AD/A-002 696

IMU SELF-ALIGNMENT TECHNIQUES

J. C. Hung, et al

Battelle Columbus Laboratories

Prepared for:

Army Missile Research, Development and Engineering Laboratory

30 September 1974

DISTRIBUTED BY:

NTIS

National Technical Information Service
U. S. DEPARTMENT OF COMMERCE

DISPOSITION INSTRUCTIONS

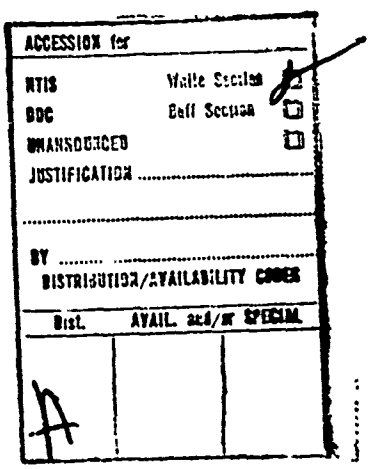
DESTROY THIS REPORT WHEN IT IS NO LONGER NEEDED. DO NOT RETURN IT TO THE ORIGINATOR.

DISCLAIMER

THE FINDINGS IN THIS REPORT ARE NOT TO BE CONSTRUED AS AN OFFICIAL DEPARTMENT OF THE ARMY POSITION UNLESS SO DESIGNATED BY OTHER AUTHORIZED DOCUMENTS.

TRADE NAMES

USE OF TRADE NAMES OR MANUFACTURERS IN THIS REPORT DOES NOT CONSTITUTE AN OFFICIAL INDORSEMENT OR APPROVAL OF THE USE OF SUCH COMMERCIAL HARDWARE OR SOFTWARE.



UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

AD/A002-696

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: RG-75-15	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): IMU SELF-ALIGNMENT TECHNIQUES	5. TYPE OF REPORT & PERIOD COVERED: Technical Report
6. PERFORMING ORG. REPORT NUMBER	
7. AUTHOR(s): J. C. Hung and H. V. White	8. CONTRACT OR GRANT NUMBER(s): DAHC04-72-A-0001 Task Order 74-208
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Battelle Columbus Laboratories, OHIO [redacted]	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: (DA) 1M263306D077 AMCMS Code 232162.55.12700
11. CONTROLLING OFFICE NAME AND ADDRESS: US Army Missile Research, Development and Engineering Laboratory US Army Missile Command Redstone Arsenal, Alabama 35809	12. REPORT DATE: 30 September 1974
	13. NUMBER OF PAGES: 65
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Platform self-alignment
Gyrocompassing algorithm

Reproduced by
NATIONAL TECHNICAL
INFORMATION SERVICE
US Department of Commerce
Springfield, VA. 22151

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report presents the results of a study on platform self-alignment performed at the Guidance and Control Directorate, US Army Missile Research, Development and Engineering Laboratory, Redstone Arsenal, Alabama.

The study was initiated to explore the latest techniques in platform self-alignment and to develop new and novel approaches which would enhance the successful application of self-alignment principles to the PERSHING II platform alignment problem with its unique accuracy and reaction time constraints.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED (65)

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Block 20 continued

A significant result of this study is the development of a new gyrocompassing algorithm which provides improved self-alignment performance.

CONTENTS

Section I. INTRODUCTION

This report documents the results of a study on inertial measurement unit (IMU) self-alignment techniques for PERSHING II. The primary concern is the fine alignment of a fixed base inertial platform where the base is subjected to ground vibration and wind buffeting.

A thorough discussion will be made on the relationships among drifts and misalignments of a platform. A gyrocompassing equation will be derived. Several concepts useful for forming self-alignment procedures will be discussed with the help of developed analytics.

A new least square regression algorithm, specially for IMU alignment, will be developed. The superiority of this algorithm will be demonstrated through theoretical analysis, experimental results, and hypothetical examples.

The scope of this study can best be seen from the Table of Contents.

Section II. IMU SELF-ALIGNMENT

An IMU can be aligned to an earth fixed coordinate system at most latitudes on the earth's surface by using the information derived from the output of the unit's sensors. This type alignment of an IMU is called leveling and gyrocompassing. Two fundamentally different approaches are used to accomplish this: (1) the gimbaled platform of the IMU is physically driven to align with the earth coordinates, and (2) the alignment is achieved analytically by determining the misalignments of platform axes with respect to the earth coordinates. The second approach has the advantage of faster gyrocompassing, but at the expense of a high speed digital computer. Our present study is centered on the second approach, namely, the "analytic gyrocompassing".

The earth fixed coordinate system adopted in this study is shown in Figure 1, where the three orthogonal axes are N, E, and A representing north, east, and azimuth, respectively. As a result of this choice of coordinates, the azimuth component of the earth rate is a negative quantity as shown in Figure 2.

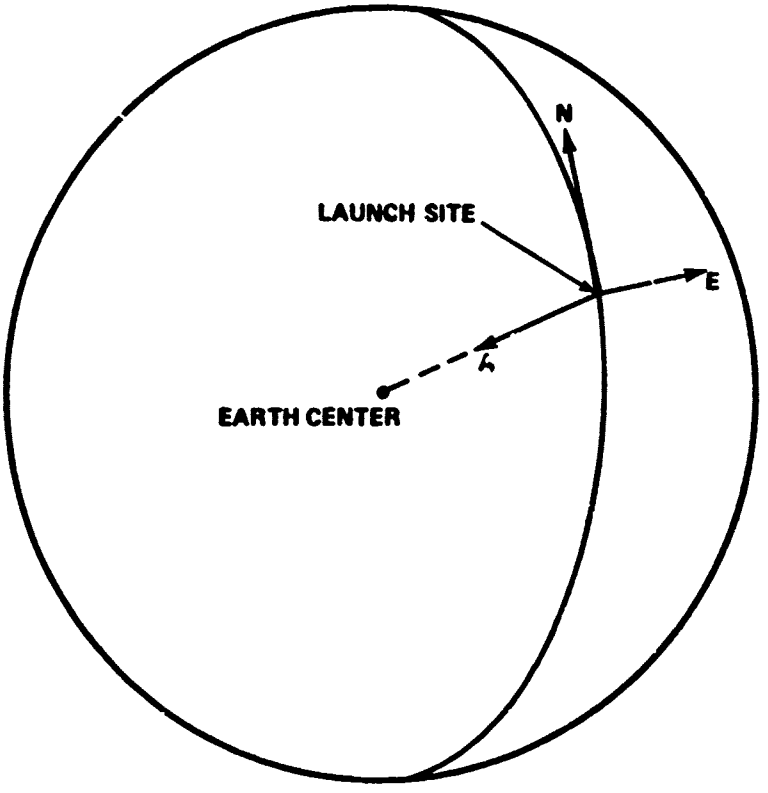


Figure 1. Earth fixed coordinates.

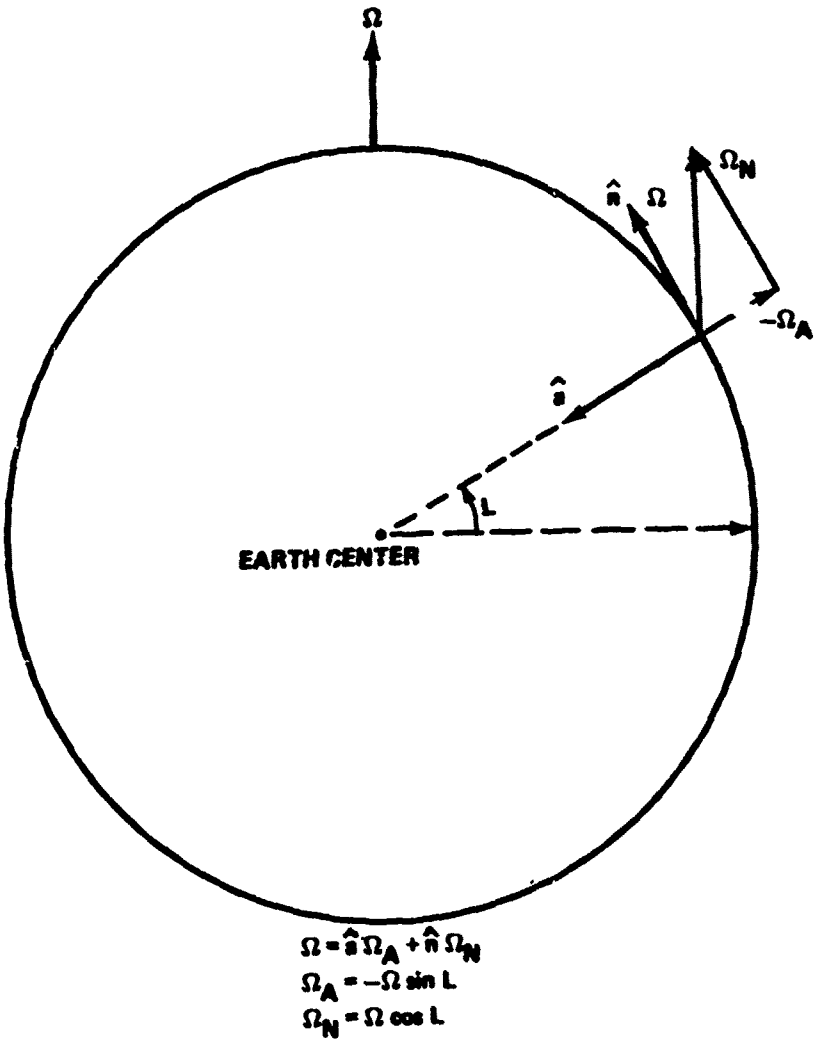


Figure 2. Earth rate components.

Figure 3 depicts the involvement of the analytic gyrocompassing. A large amount of data, obtained from the outputs of platform sensors, is reduced to a set of parameters through a chosen data reduction technique. Then, the platform's azimuth is determined from these parameters using a gyrocompassing equation.

It is obvious that better hardware allows more accurate gyrocompassing. For a given hardware system, different gyrocompassing procedures can be formed by different combinations of basic concepts. Thus the accuracy of gyrocompassing depends on several factors:

1) Platform hardware

2) Gyrocompassing procedure

3) Gyrocompassing equation

4) Data reduction algorithm

5) Computer dependent errors.

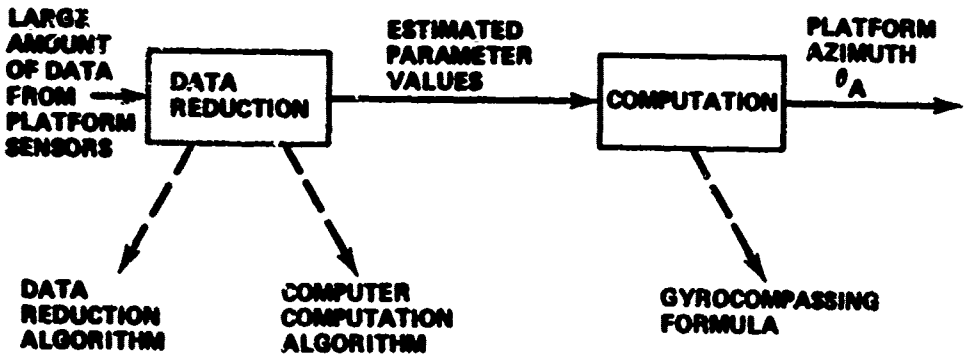


Figure 3. Analytic gyrocompassing.

Figure 4 shows a general model of a platform with error sources indicated. Each solid line represents a hardware connection while each dashed line represents the path of a sensed signal.

The "electronic and network" block can be implemented for various purposes such as maintaining the platform at a specific orientation, compensating for errors, and improving the platform dynamics. Our present purpose is to align the platform coordinates to the earth fixed coordinates.

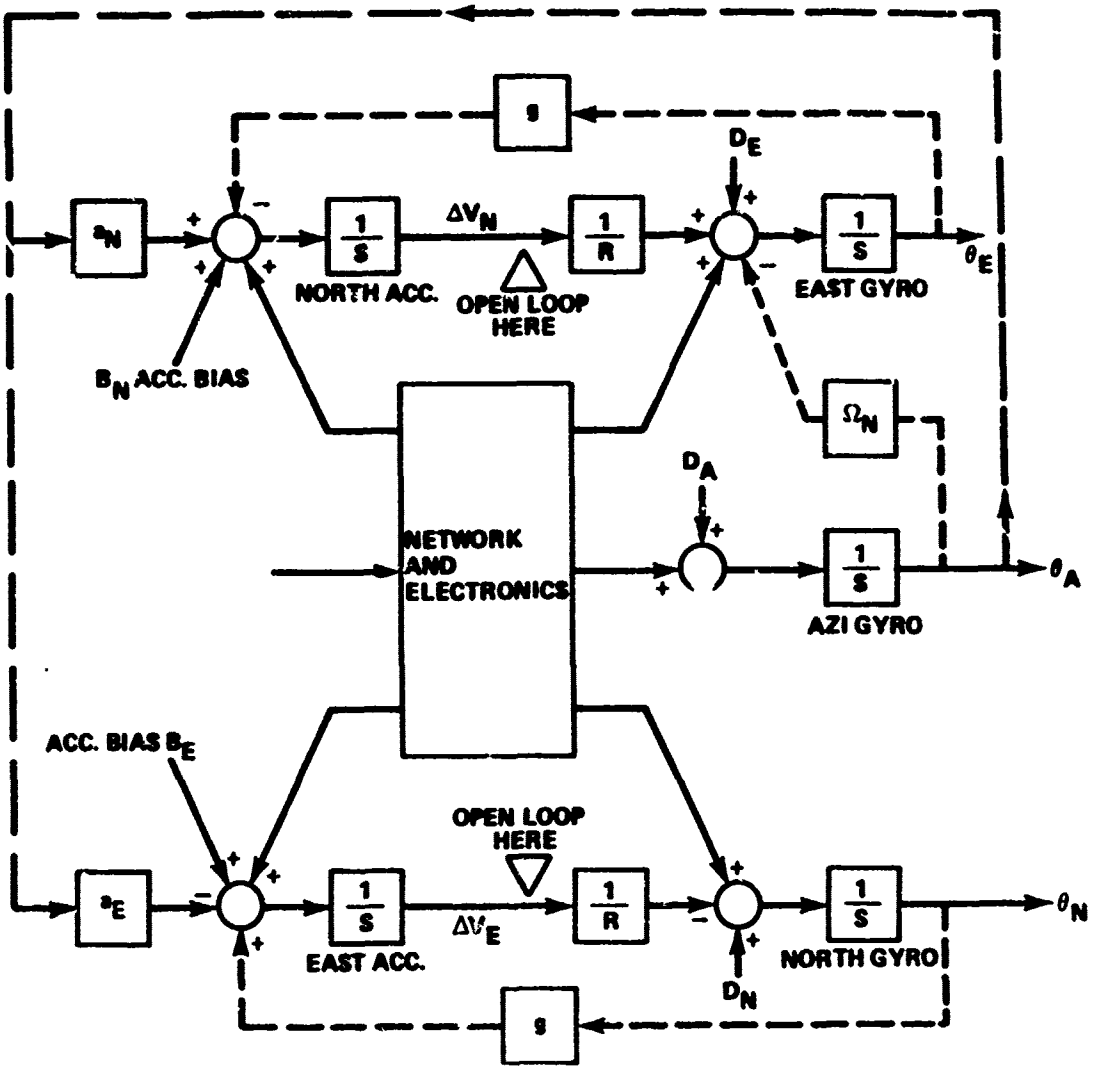


Figure 4. Platform alignment model.

Section III. PLATFORM DRIFTS [1]

The analytic model needed for data reduction and the gyrocompassing equation needed for azimuth determination are derived from the drift characteristics of the platform. Therefore understanding the drift characteristics is a prerequisite to the development of gyrocompassing techniques.

For a gimbaled platform, the platform axes are slaved to the gyros. The time constants of platform servos are usually much smaller than the gyrocompassing time (on the order of milliseconds versus minutes). Hence the gyro drift contributes instantaneously to the platform drift of the same amount. Thus the terms "platform drift" and "gyro torquing rate" become synonymous.

1. Kinematics of Platform Drift

Consider a platform which has been coarsely aligned to the earth fixed coordinates. The deterministic torquing rate for each gyro consists of the self-axis earth rate component, the cross-axis earth rate component, and the gyro drift.

Let θ_N = misalignment about north axis

θ_E = misalignment about east axis

θ_A = misalignment about azimuth axis

D_N = north gyro drift

D_E = east gyro drift

D_A = azimuth gyro drift rate

Ω = earth rate

L = latitude of launchsite

$\Omega_N = \Omega \cos L$ = north component of earth rate

$\Omega_A = -\Omega \sin L$ = azimuth component of earth rate

K_N = north gyro torquer scale factor error

K_A = azimuth gyro torquer scale factor error.

The torquing rate for each gyro is obtained as follows:

for the north gyro,

$$\dot{\theta}_N = \Omega_N + D_N - \Omega_A \sin\theta_E - \Omega_N(1 - \cos\theta_A \cos\theta_E) + K_N \Omega_N \qquad (1)$$

where $\Omega_A \sin\theta_E$ = cross-axis earth rate due to misalignment

$\Omega_N(1 - \cos\theta_A \cos\theta_E)$ = change of self-axis earth rate due to misalignment

$K_N \Omega_N$ = rate due to torquer scale factor error;

for the east gyro,

$$\dot{\theta}_E = D_E + \Omega_A \sin\theta_N - \Omega_N \sin\theta_A \qquad (2)$$

where $\Omega_A \sin\theta_N - \Omega_N \sin\theta_A$ = cross-axis earth rates due to misalignment;

for the azimuth gyro,

$$\dot{\theta}_A = \Omega_A + D_A + \Omega_N \sin\theta_E - \Omega_A(1 - \cos\theta_N \cos\theta_E) + K_A \Omega_A \qquad (3)$$

where $\Omega_N \sin\theta_E$ = cross-axis earth rate due to misalignment

$\Omega_A(1 - \cos\theta_N \cos\theta_E)$ = change of self-axis earth rate due to misalignment

$K_A \Omega_A$ = rate due to torquer scale factor error.

From Equations (1), (2), and (3) the non-nominal parts of the torquing rates for north, east, and azimuth gyros (or, equivalently, the drifts for north, east, and azimuth axes of the platform) are, respectively,

$$D_N' = D_N - \Omega_A \sin\theta_E - \Omega_N(1 - \cos\theta_A \cos\theta_E) + K_N \Omega_N \qquad (4)$$

$$D_E' = D_E + \Omega_A \sin\theta_N - \Omega_N \sin\theta_A \qquad (5)$$

$$D_A' = D_A + \Omega_N \sin\theta_E - \Omega_A(1 - \cos\theta_N \cos\theta_E) + K_A \Omega_A \quad . \qquad (6)$$

If the misalignment θ_N, θ_E, and θ_A are sufficiently small, small angle approximations for sine and cosine functions can be used. Under

this condition, Equations (4), (5), and (6) reduce to

$$D_N' = D_N - \Omega_A \theta_E + K_N \Omega_N \tag{7}$$

$$D_E' = D_E - \Omega_N \theta_A + \Omega_A \theta_N \tag{8}$$

$$D_A' = D_A + \Omega_N \theta_E + K_N \Omega_A \quad . \tag{9}$$

Monitoring values of calibrated gyro torquing currents provide a way of determining the platform drifts.

2. Time Functions of Drifts

Equations (7), (8), and (9) show that the drifts along three platform axes are coupled together by the misalignments θ_N, θ_E, and θ_A. Understanding this coupling effect is important to accurate gyrocompassing.

It is reasonable to assume that during the period of gyrocompassing, θ_A, the azimuth misalignment is constant. With this in mind, Equations (7) and (8) can be further developed into

$$\begin{aligned} D_N' &= D_N - \Omega_A \theta_E + K_N \Omega_N \\ &= D_N - \Omega_A \left(\theta_{E0} + \int_0^t D_E'(\tau)\, d\tau \right) + K_N \Omega_N \\ &= \underbrace{D_N - \Omega_A \theta_{E0} + K_N \Omega_N}_{D_{N0}'} - \Omega_A \int_0^t D_E'(\tau)\, d\tau \end{aligned} \tag{10}$$

and

$$\begin{aligned} D_E' &= D_E - \Omega_N \theta_A + \Omega_A \theta_N \\ &= D_E - \Omega_N \theta_A + \Omega_A (\theta_{N0} + \int_0^t D_N'(\tau)\, d\tau \\ &= \underbrace{D_E - \Omega_N \theta_A + \Omega_A \theta_{N0}}_{D_{E0}'} + \Omega_A \int_0^t D_N'(\tau)\, d\tau \quad . \end{aligned} \tag{11}$$

By defining

$$\left.\begin{aligned} D'_{EO} &= D_E - \Omega_N \theta_A + \Omega_A \theta_{NO} \simeq D_E - \Omega_N \theta_A \\ D'_{NO} &= D_N - \Omega_A \theta_{EO} + K_N \Omega_N \end{aligned}\right\} , \tag{12}$$

which represent the initial values of the drift for east and north axes, the two drift equations become

$$D'_N + \Omega_A \int_0^t D'_E(\tau)d\tau = D'_{NO} \tag{13}$$

$$D'_E - \Omega_A \int_0^t D'_N(\tau)d\tau = D'_{EO} \quad . \tag{14}$$

The time functions $D'_N(t)$ and $D'_E(t)$ can be solved from Equations (13) and (14) using transform method. Taking the Laplace transform of both equations,

$$D'_N(s) + \frac{\Omega_A}{s} D'_E(s) = \frac{D'_{NO}}{s} \tag{15}$$

$$D'_E(s) - \frac{\Omega_A}{s} D'_N(s) = \frac{D'_{EO}}{s} \quad . \tag{16}$$

Solving for $D'_N(s)$ and $D'_E(s)$ yields

$$D'_N(s) = D'_{NO} \frac{s}{s^2 + \Omega_A^2} - D'_{EO} \frac{\Omega_A}{s^2 + \Omega_A^2} \tag{17}$$

$$D'_E(s) = D'_{EO} \frac{s}{s^2 + \Omega_A^2} + D'_{NO} \frac{\Omega_A}{s^2 + \Omega_A^2} \quad . \tag{18}$$

By taking the inverse Laplace transform of Equations (17) and (18), the corresponding time functions are, respectively,

$$D'_N(t) = D'_{NO} \cos \Omega_A t - D'_{EO} \sin \Omega_A t \tag{19}$$

$$D'_E(t) = D'_{EO} \cos \Omega_A t + D'_{NO} \sin \Omega_A t \quad . \tag{20}$$

In matrix form,

$$\begin{bmatrix} D'_N(t) \\ D'_E(t) \end{bmatrix} = \begin{bmatrix} \cos \Omega_A t & -\sin \Omega_A t \\ \sin \Omega_A t & \cos \Omega_A t \end{bmatrix} \begin{bmatrix} D'_{N0} \\ D'_{E0} \end{bmatrix} \tag{21}$$

which shows that the vector drift at any time is a rotation of angle $\Omega_A t$ from its initial vector drift.

For small value of $\Omega_A t$,

$$\left.\begin{aligned} \cos \Omega_A t &\simeq 1 - \frac{\Omega_A^2 t^2}{2} \\ \sin \Omega_A t &\simeq \Omega_A t \end{aligned}\right\} . \tag{22}$$

Then Equations (17) and (18) can be approximated by

$$D'_N(t) = D'_{N0} - D'_{E0}\, \Omega_A t - D'_{N0} \frac{\Omega_A^2 t^2}{2} \tag{23}$$

$$D'_E(t) = D'_{E0} + D'_{N0}\, \Omega_A t - D'_{E0} \frac{\Omega_A^2 t^2}{2} . \tag{24}$$

If the second order terms are negligible, Equations (23) and (24) can further be approximated by

$$D'_N(t) = D'_{N0} - D'_{E0}\, \Omega_A t \tag{25}$$

$$D'_E(t) = D'_{E0} + D'_{N0}\, \Omega_A t . \tag{26}$$

3. Time Functions of Misalignments

With the help of Equations (19) and (20), the misalignments $\theta_N(t)$ and $\theta_E(t)$ can be determined as follows:

$$\theta_N(t) = \theta_{NO} + \int_0^t D'_N(\tau)\,d\tau$$

$$= \theta_{NO} + \int_0^t (D'_{NO} \cos \Omega_A\tau - D'_{EO} \sin \Omega_A\tau)\,d\tau$$

$$= \theta_{NO} + \frac{D'_{NO}}{\Omega_A} \sin \Omega_A t + \frac{D'_{EO}}{\Omega_A} \cos \Omega_A t - \frac{D'_{EO}}{\Omega_A} \tag{27}$$

$$\theta_E(t) = \theta_{EO} + \int_0^t D'_E(\tau)\,d\tau$$

$$= \theta_{EO} + \int_0^t (D'_{EO} \cos \Omega_A\tau + D'_{NO} \sin \Omega_A\tau)\,d\tau$$

$$= \theta_{EO} + \frac{D'_{EO}}{\Omega_A} \sin \Omega_A t - \frac{D'_{NO}}{\Omega_A} \cos \Omega_A t + \frac{D'_{NO}}{\Omega_A} \quad . \tag{28}$$

Section IV. GYROCOMPASSING EQUATION

Gyrocompassing, the determination of azimuth misalignment, requires the knowledge of drifts and misalignments along the north and east platform axes. A gyrocompassing equation and its accuracy are discussed here.

1. The Equation

By solving Equation (8) for $\theta_A(t)$, the azimuth misalignment at any time t is obtained as

$$\theta_A(t) = \frac{D_E - D'_E(t) + \Omega_A\, \theta_N(t)}{\Omega_N} \quad . \tag{29}$$

Substituting the details of $D'_E(t)$ and $\theta_N(t)$ from Equations (18) and (25) into Equation (29), all sine and cosine terms cancel. The result is the "gyrocompassing equation" sought,

$$\theta_A(t) = \frac{D_E - D'_{E0} + \Omega_A\, \theta_{N0}}{\Omega_N} \quad . \tag{30}$$

Intuitively, it can also be said that Equation (30) comes directly from Equation (29) since

$$\theta_A(t) \simeq \theta_{A0} = \frac{D_E - D'_{E0} + \Omega_A\, \theta_{N0}}{\Omega_N} \quad .$$

2. Gyrocompassing Accuracy

The ultimate gyrocompassing accuracy is limited by the following uncertainties:

ΔD_E = East gyro drift uncertainty

$\Delta D'_E$ = Uncertainty in platform drift about its east axis

$\Delta\theta_{N0}$ = Uncertainty in the initial platform misalignment about its north axis.

From Equation (30) the uncertainty in gyrocompassing is obtained as

$$\Delta\theta_A = \frac{\Delta D_E - \Delta D'_E + \Omega_A\, \Delta\theta_{N0}}{\Omega_N} \quad . \tag{31}$$

Since the sign of each individual uncertainty is not known, an upper bound of the gyrocompassing error is given by

$$|\Delta\theta_A| \leq \frac{|\Delta D_E' - \Delta D_E| + |\Omega_A \Delta\theta_{NO}|}{\Omega_N} \quad . \tag{32}$$

In Equation (32), $\Delta\theta_A$ and $\Delta\theta_{NO}$ are in radians while ΔD_E, $\Delta D_E'$, Ω_A, and Ω_N have the same unit. If $\Delta\theta_A$ and $\Delta\theta_{NO}$ are in arcseconds, Equation (32) should be replaced by

$$|\Delta\theta_A| \leq 206{,}280 \frac{|\Delta D_E' - \Delta D_E|}{\Omega_N} + |\Delta\theta_{NO} \tan L| \tag{33}$$

where L is the launchsite latitude.

Consider an example where

$$L = 45 \text{ degrees}$$

$$\Delta D_E' - \Delta D_E = 0.003 \text{ deg/hr}$$

$$\Delta\theta_{NO} = 2 \text{ arcsec} \quad .$$

Since $\Omega = 15$ degrees/hour,

$$\Omega_N = \Omega \cos L = 10.61$$

$$\tan L = 1 \quad .$$

Equation (33) gives

$$|\Delta\theta_A| \leq \left|206{,}280 \times \frac{0.003}{10.61}\right| + 2 = 58.3 + 2 = 60.3 \text{ arcsec} \quad .$$

Notice that, in this example, the platform east axis drift uncertainty contributes most of the error in gyrocompassing.

Section V. CONCEPTS FOR SELF-ALIGNMENT

This section presents a discussion of several useful concepts which can be chosen to form different IMU self-alignment procedures.

1. Gyro Drift Determination

Determination of gyro drifts using the information within the atform system is also called "autobiasing." Here we shall be con- .ned with drifts about north and east axes. There are two different ..ethods of autobiasing gyros, namely, the "closed-loop method" and the "open-loop method." The choice between the two depends on the relative uncertainty between the gyro torquer scale factor error and the accelerometer scale factor error.

Referring to Figure 4, the platform alignment system consists of two Schuler loops. For the closed-loop method, both Schuler loops are closed and gyros are torqued at the rates given by Equations (1), (2), and (3). Under the fine alignment condition, platform is sufficiently level such that small angle approximations for trigonometric functions are satisfactory. Therefore Equations (7) and (8) give the non-nominal torquing rate for the north and east gyros. In general, there are biases in accelerometers, so the terms $-\Omega_A\theta_E$ and $\Omega_A\theta_N$ may not be small. However, in all practical cases, the biases are known. Therefore their effect on platform drift is known. Thus it can be said that the difference between $-\Omega_A\theta_E$ and the corresponding accelerometer bias effect is small, and between $\Omega_A\theta_N$ and its corresponding accelerometer bias effect is also small. Under this condition, Equations (7) and (8) are further reduced to

$$D_N' \simeq D_N + K_N\Omega_N \tag{34}$$

$$D_E' \simeq D_E - \Omega_N\theta_A \quad . \tag{35}$$

The quantities D_N' and D_E' are obtained by measuring the torquing currents of north and east gyros. Assuming that Ω_N is known, the north gyro drift D_N can be accurately determined if K_N, the torquer scale factor error, is known. However, the uncertainty in $\Omega_N\theta_A$ is, in general, so large that there is no way to accurately determine the east gyro drift D_E from D_E'.

Often the knowledge of K_N is not available to the degree of precision desired. Under this condition, accurate and rapid determination of D_N from the closed-loop information is difficult.

The effect of uncertainty in north gyro torquer scale factor error can be eliminated entirely by not torquing the north axis of the platform physically. Instead, an analytically torqued north axis is maintained in the computer by on-line computation. This method is called "zero-torquing measurement." Zero-torquing is accomplished by opening Schuler loops at places indicated in Figure 4. Therefore the method is an "open-loop method." Under this condition, platform level is not maintained, so accelerometers receive larger inputs. Thus the uncertainty in the accelerometer's scale factor error becomes more important.

In the open-loop method, measurements are taken at outputs of both accelerometers. From the measurements, D_N' and D_E' are determined. Because of zero-torquing, K_N plays no part in drift determination, so Equation (34) becomes

$$D_N' \simeq D_N \quad , \tag{36}$$

which is an attractive way to determine D_N. However, D_E' is still given by Equation (35) where separation of D_E from $-\Omega_N \theta_A$ is difficult.

To conclude, it is seen that whether the closed-loop method or open-loop method is used to determine gyro drifts, only the north gyro drift can be accurately determined. Reference 1 contains several numerical examples to illustrate this phenomenon. The technique of determining D_N' and D_E' from the measurements for the open-loop method will be discussed in detail in Sections VI and VII.

2. Two-Position Gyrocompassing

Recall from Equation (30) that an accurate gyrocompassing can be achieved only if an accurate determination of the east gyro drift D_E can be obtained. Since accurate drift determination can be made only for north gyro, a two-position scheme can be devised to take this advantage. The scheme may consist of the following steps:

a) Autobiasing the north gyro to determine D_N

b) Slewing the platform approximately 90 degrees so that the north gyro becomes an east gyro, and the original east gyro becomes a south gyro

c) Gyrocompassing is performed with the present east gyro whose drift has been accurately determined, enabling an accurate azimuth alignment.

After slewing, the south gyro is in a favorable position for accurate drift determination. The autobiased south gyro will then be ready for in-flight navigation.

3. Off-Set Self-Alignment

It is possible to achieve platform self-alignment with the platform coarsely leveled but without requiring that the platform axes be physically coarsely aligned to north and east. Instead, the north and east, which are obtained from an analytic coarse alignment, are analytically maintained in the computer. The fine alignment is then achieved by determining the misalignments between the computer north and east and the true north and east.

Referring to Figure 5, α denotes the off-set of the platform level coordinates from the computer's level coordinates, and θ_A is the azimuth misalignment to be determined. Under the off-set condition, the following quantities are first obtained:

V_{N_P} and V_{E_P} — Velocities along the platform's north and east axes, obtained by integrating the outputs of north and east accelerometers

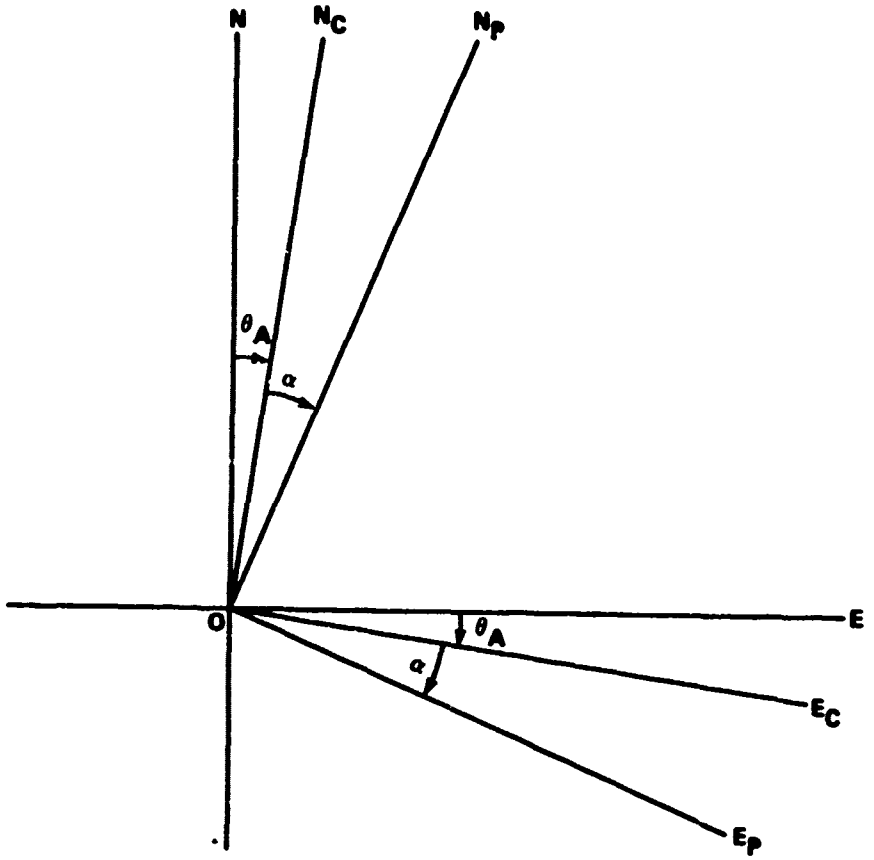
α — The off-set angle, obtained by a certain coarse alignment procedure, say, BATH.

Next, the velocities along the computer north and east axes are computed from

$$\left.\begin{aligned} V_{N_C} &= V_{N_P} \cos\alpha - V_{E_P} \sin\alpha \\ V_{E_C} &= V_{E_P} \cos\alpha + V_{N_P} \sin\alpha \end{aligned}\right\} . \qquad (37)$$

Finally, a fine alignment technique can be chosen for performing the fine alignment between the computer axes and the earth coordinates.

During the fine alignment, platform can either be torqued at earth rate, or be torqued about the azimuth axis at the azimuth component of the earth rate. The merit of not torquing the level axes is to avoid the torquer scale factor uncertainties, which have been discussed. The information of the torqued coordinates is already in the computer since this information is needed to torque the platform in the first place.



E, N – EARTH EAST AND NORTH
E_C, N_C – COMPUTER EAST AND NORTH
E_P, N_P – PLATFORM EAST AND NORTH

Figure 5. Off-set self-alignment.

4. Large Angle Self-Alignment

In the case of "large angle self-alignment," the platform is coarsely leveled so the small angle approximations for θ_N and θ_E are valid. The azimuth axis is torqued at Ω_A, the azimuth component of earth rate. But the azimuth misalignment θ_A is not small enough to allow the use of small angle approximation.

Under this condition, a good approximation for platform drift can be obtained from Equations (1) and (2) as

$$D_N' = D_N - \Omega_A \theta_E + \Omega_N \cos \theta_A \qquad (38)$$

$$D'_E = D_E + \Omega_A \theta_N - \Omega_N \sin \theta_A \quad . \tag{39}$$

Equations (38) and (39) can be solved for $\cos \theta_A$ and $\sin \theta_A$, respectively. Thus $\tan \theta_A$ can also be obtained. During the self-alignment period, the variation in azimuth misalignment is small, so it is assumed that $\theta_A = \theta_{A0}$. The resulting gyrocompassing equation is therefore given by

$$\theta_A = \theta_{A0} = \tan^{-1} \frac{\Omega_A \theta_{N0} + (D_E - D'_{E0})}{\Omega_A \theta_{E0} - (D_N - D'_{N0})} \quad . \tag{40}$$

The ambiguity of double values can be resolved by solving θ_A from Equation (39) also, and compare its sign to that determined by Equation (40). The determination of θ_{N0}, θ_{E0}, D'_{N0}, and D'_{E0} will be discussed in Sections VI and VII.

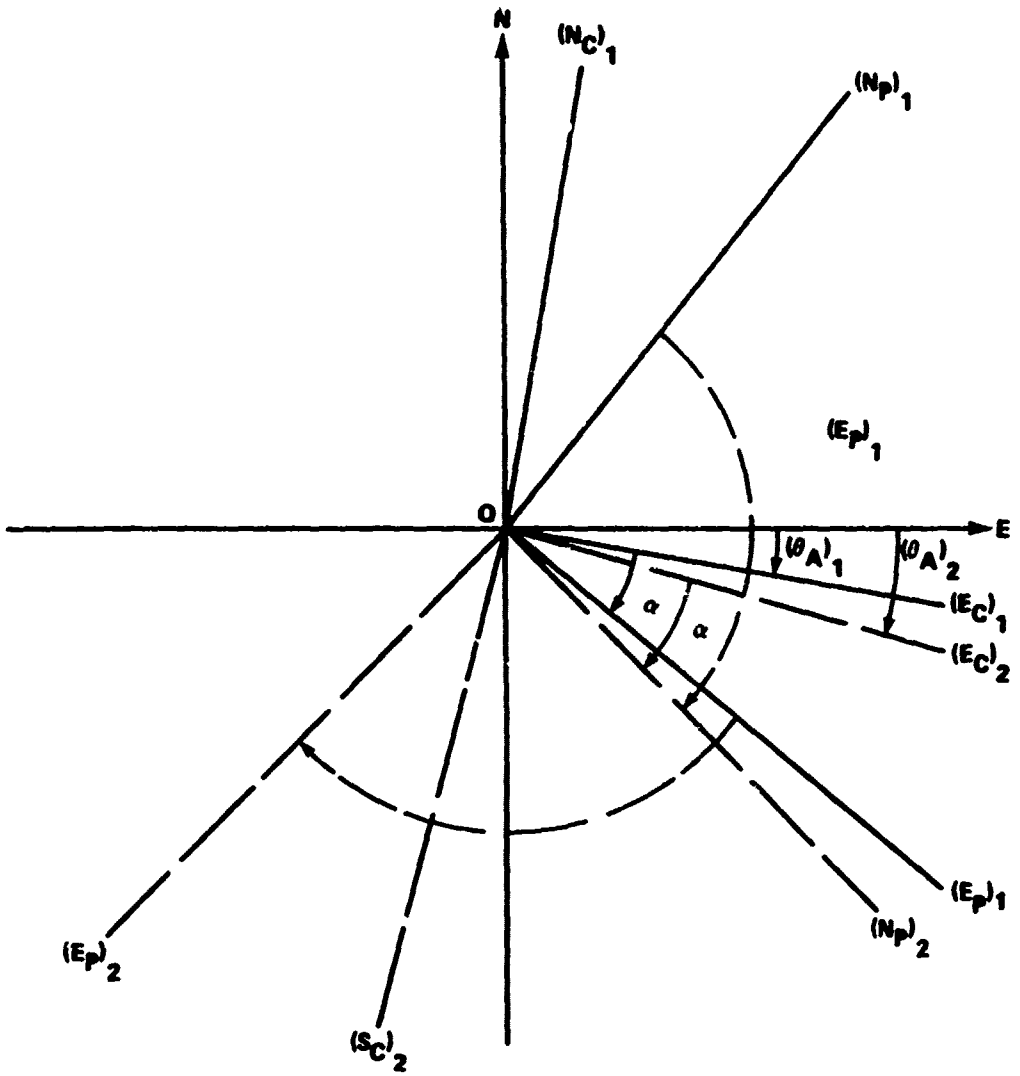
One may wonder why θ_A is not determined directly from either Equation (38) or (39) by taking the inverse of cosine or sine function. The reason is that tangent function possesses steeper slopes, enabling a more accurate determination of θ_A.

Notice that the large azimuth angles for which this method is intended cannot be arbitrarily large. They must be within the limits that Equations (1) and (2) are valid.

5. A Typical Self-Alignment Procedure

By combining a few of the aforementioned concepts, a typical self-alignment procedure can be developed. As an example we may have a "two-position off-set zero-torquing self-alignment." Figure 6 shows the coordinates involved in this method. In Figure 6 N and E are earth's north and east; N_C, E_C, and S_C are the north, east, and south known to the computer; and N_P and E_P are north and east of the platform. The 90-degree slewing for the second position is indicated by dashed arcs. The slewing is not required to be precise.

The required alignment steps for this example can best be described with the help of an activity flow diagram shown in Figure 7. If the time for achieving BATH is shorter than a coarse alignment slewing, the total alignment time can be reduced when the off-set self-alignment concept is employed. This is because the physical coarse alignment takes time to achieve, while the off-set technique 90-degrees slewing can be very rapid without worrying about its accuracy.



E, N – EARTH EAST AND NORTH
E_C, N_C, S_C – COMPUTER EAST, NORTH AND SOUTH
E_P, N_P – PLATFORM EAST AND NORTH

Figure 6. Coordinates for a 2-position off-set self-alignment.

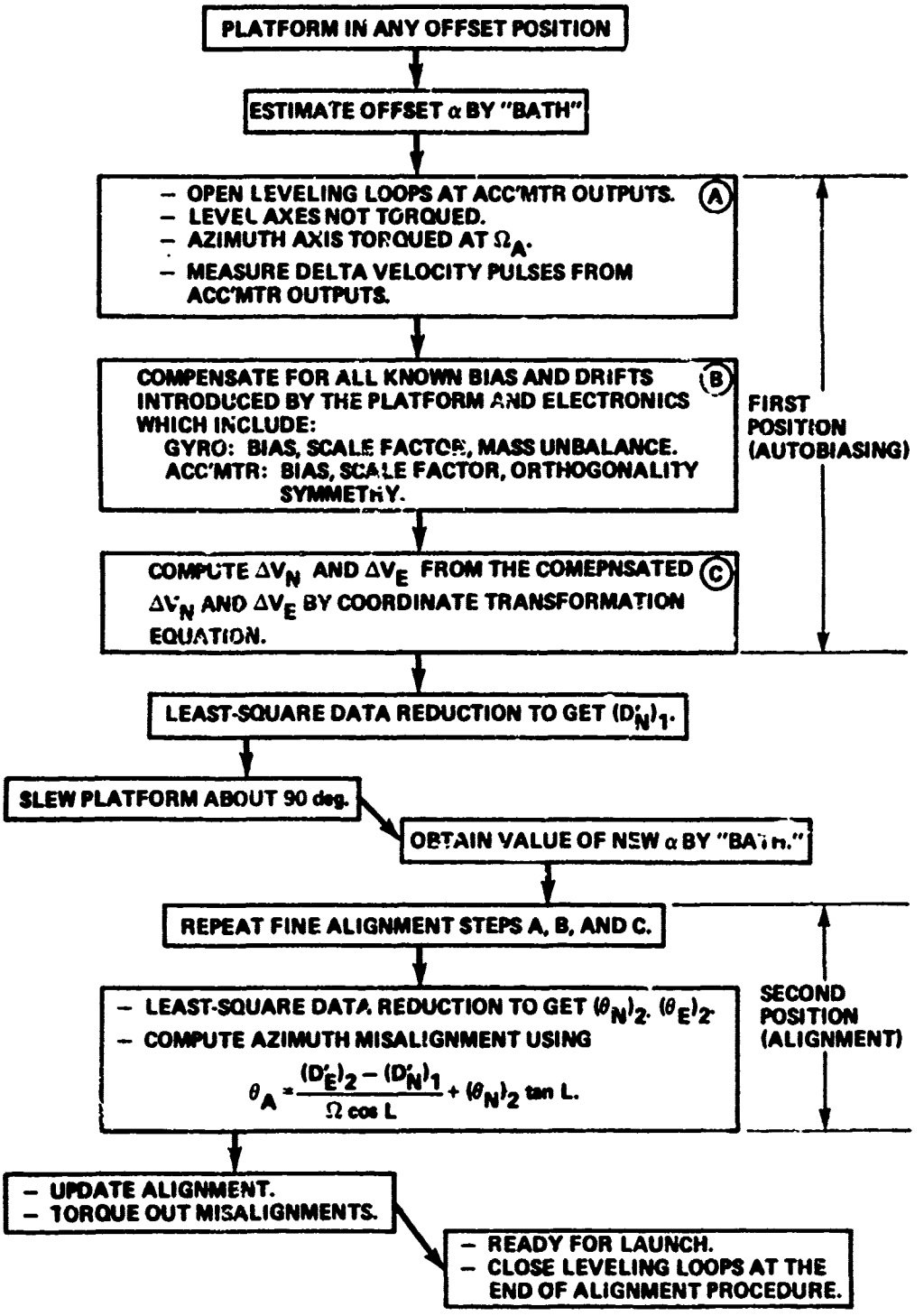


Figure 7. Two-position off-set self-alignment.

Section VI. STATE ESTIMATION FOR SELF-ALIGNMENT

1. Platform Alignment State Vector

Platform leveling and gyrocompassing amount to the determination of misalignments θ_N, θ_E, and θ_A which, in turn, require the knowledge of D'_N and D'_E. A preferred procedure is to first determine θ_N, θ_E, D'_N, and D'_E from the platform's sensor outputs. The azimuth θ_A can then be determined using the gyrocompassing Equation (30). In this procedure the desired self-alignment state vector $\underline{x}$ consists of four elements:

$$\underline{x} = \begin{bmatrix} \theta_N \\ \theta_E \\ D'_N \\ D'_E \end{bmatrix} . \tag{40}$$

2. Zero-Torquing Measurement Equation

Consider the case of zero-torquing fine alignment where the state vector is determined from the output data of accelerometers. For a sufficiently short fine alignment time the equations for misalignments, Equations (27) and (28), can be approximated by their Taylor series expansion up to the second power of t; i.e.,

$$\theta_N(t) = \theta_{NO} + D'_{NO}t - \frac{1}{2} D'_{EO} \Omega_A t^2 \tag{41}$$

$$\theta_E(t) = \theta_{EO} + D'_{EO}t + \frac{1}{2} D'_{NO} \Omega_A t^2 . \tag{42}$$

Notice that, since the north gyro is not torqued, the drift term D'_{NO} includes the north component of the earth rate. The drifts and misalignments transform into accelerations via tilts of north and east accelerometers, giving

$$a_N = -g \times \sin \theta_E \simeq -g\theta_E \tag{43}$$

$$a_E = g \times \sin \theta_N \simeq g\theta_N \tag{44}$$

where g is the gravitational acceleration. Substituting Equations (41) and (42) into Equations (43) and (44) gives

$$a_N = -g\theta_{EO} - gD'_{EO}t - \frac{g}{2} D'_{NO} \Omega_A t^2 \qquad (45)$$

$$a_E = g\theta_{NO} + gD'_{NO}t - \frac{g}{2} D'_{EO} \Omega_A t^2 \quad . \qquad (46)$$

Integrating Equations (45) and (46) from 0 to t gives the changes of velocities during that period as

$$V_N(t) = -g\theta_{EO}t - \frac{g}{2} D'_{EO}t^2 - \frac{g}{6} D'_{NO} \Omega_A t^3 \qquad (47)$$

$$V_E(t) = g\theta_{NO}t + \frac{g}{2} D'_{NO}t^2 - \frac{g}{6} D'_{EO} \Omega_A t^3 \quad . \qquad (48)$$

Let $\underline{V} = [V_N, V_E]^t$ be the measurement vector, the matrix form of Equations (47) and (48) is

$$\begin{bmatrix} V_N(t) \\ V_E(t) \end{bmatrix} = \begin{bmatrix} 0 & -gt - \frac{g}{6} \Omega_A t^3 & -\frac{g}{2} t^2 \\ gt & 0 & \frac{g}{2} t & -\frac{g}{6} \Omega_A t^3 \end{bmatrix} \begin{bmatrix} \theta_{NO} \\ \theta_{EO} \\ D'_{NO} \\ D'_{EO} \end{bmatrix} \quad . \qquad (49)$$

Equations (47) and (48), or Equation (49) are the measurement equation desired.

Grouping Equations (19), (20), (27), and (28) together, the state vector at any time is related to its initial value by

$$\begin{bmatrix} \theta_N \\ \theta_E \\ D'_N \\ D'_E \end{bmatrix} = \begin{bmatrix} 1 & 0 & \frac{\sin \Omega_A t}{\Omega_A} & \frac{\cos \Omega_A t - 1}{\Omega_A} \\ 0 & 1 & \frac{1 - \cos \Omega_A t}{\Omega_A} & \frac{\sin \Omega_A}{\Omega_A} \\ 0 & 0 & \cos \Omega_A t & -\sin \Omega_A t \\ 0 & 0 & \sin \Omega_A t & \cos \Omega_A t \end{bmatrix} \begin{bmatrix} \theta_{NO} \\ \theta_{EO} \\ D'_{NO} \\ D'_{EO} \end{bmatrix} \quad . \qquad (50)$$

Equations (49) and (50) provide a way for determining the desired state vector from the integrated measurement data.

It may be asked why the state vector is not determined using Equations (45) and (46) as measurement equations. The answer is that the integration reduces errors caused by the quantization effect of the accelerometer output.

In practice, discrete measurements are made at time $t = \tau k$ for $k = 1 \sim N$, and τ is the sampling period. Substituting the discrete time into Equations (49) and (50) gives

$$\begin{bmatrix} V_N(k) \\ V_E(k) \end{bmatrix} = \begin{bmatrix} 0 & -g\tau k & -\frac{g}{6}\Omega_A\tau^3k^3 & -\frac{g}{2}\tau^2k^2 \\ g\tau k & 0 & \frac{g}{2}\tau^2k^2 & -\frac{g}{6}\Omega_A\tau^3k^3 \end{bmatrix} \begin{bmatrix} \theta_{NO} \\ \theta_{EO} \\ D'_{NO} \\ D'_{EO} \end{bmatrix} \tag{51}$$

and

$$\begin{bmatrix} \theta_N(k) \\ \theta_E(k) \\ D'_N(k) \\ D'_E(k) \end{bmatrix} = \begin{bmatrix} 1 & 0 & \frac{\sin \Omega_A\tau k}{\Omega_A} & \frac{\cos \Omega_A\tau k - 1}{\Omega_A} \\ 0 & 1 & \frac{1 - \cos \Omega_A\tau k}{\Omega_A} & \frac{\sin \Omega_A\tau k}{\Omega_A} \\ 0 & 0 & \cos \Omega_A\tau k & -\sin \Omega_A\tau k \\ 0 & 0 & \sin \Omega_A\tau k & \cos \Omega_A\tau k \end{bmatrix} \begin{bmatrix} \theta_{NO} \\ \theta_{EO} \\ D'_{NO} \\ D'_{EO} \end{bmatrix} . \tag{52}$$

If the total measurement time is sufficiently small,

$$\left.\begin{aligned} \sin \Omega_A\tau k &\simeq \Omega_A\tau k \\ \cos \Omega_A\tau k &\simeq 1 \end{aligned}\right\} , \tag{53}$$

then an approximation for Equation (52) is

$$\begin{bmatrix} \theta_N(k) \\ \theta_E(k) \\ D'_N(k) \\ D'_E(k) \end{bmatrix} = \begin{bmatrix} 1 & 0 & \tau k & 0 \\ 0 & 1 & 0 & \tau k \\ 0 & 0 & 1 & -\Omega_A\tau k \\ 0 & 0 & \Omega_A\tau k & 1 \end{bmatrix} \begin{bmatrix} \theta_{NO} \\ \theta_{EO} \\ D'_{NO} \\ D'_{EO} \end{bmatrix} . \tag{54}$$

3. Estimation Techniques

When the environment is ideal, where disturbance and noise are not present, measurement of $V_N(t)$ and $V_E(t)$ at two different values of t is sufficient for determining the state vector. In reality, the measurements are contaminated by noise due to ground vibration, wind buffeting, instrument noise, and other random disturbances. Under this condition, accurate determination of the state vector demands the use of a large number of redundant measurements in conjunction with a statistical estimation technique.

Two well known approaches of statistical estimation techniques applicable to platform self-alignment are the least square regression approach and the Kalman filtering approach. A comparison of the two approaches is given as follows:

a) Least Square Regression - This approach does not make use of statistical properties of the noise, if available. The associated data reduction process can be made either sequential or batch. If the desired number of state estimations is much less than the number of measurements, the least square regression algorithm requires less computer time and smaller computer memory as compared to Kalman filtering [4].

b) Kalman Filtering [5] - The Kalman filtering algorithm has a built-in provision for taking advantage of known second order statistics. The associated data reduction process is sequential, which generates an estimate of the state from each measurement.

We choose the least square regression approach for the platform self-alignment. The choice is based on two facts: first, knowledge of the statistics of the noise is not good enough to enjoy the merit of Kalman filtering; and secondly, the required number of state estimations is far less than the number of measurements so least square regression approach is superior in the required computer time and memory.

In Section VII a new least square regression algorithm, tailored to our platform self-alignment application, is presented in detail.

Section VII. A NEW LEAST SQUARE ALGORITHM

1. Measurement Equations

Least square regression method requires a special form for measurement equations. Rearranging Equation (51) and adding measurement noise to it, we get

$$V_N(k) = A_1 k + A_2 k^2 + A_N k^3 + n_N(k) \tag{55}$$

$$V_E(k) = A_3 k + A_4 k^2 + A_E k^3 + n_E(k) \tag{56}$$

where $n_N(k)$ and $n_E(k)$ are additive noise, $k = 1 \sim N$, and

$$\left.\begin{aligned} A_1 &= -g\theta_{E0}\tau \\ A_2 &= -\frac{g}{2} D'_{E0}\tau^2 \\ A_N &= -\frac{g}{6} D'_{N0}\,\Omega_A\tau^3 \end{aligned}\right\} \tag{57}$$

and

$$\left.\begin{aligned} A_3 &= g\theta_{N0}\tau \\ A_4 &= \frac{g}{2} D'_{N0}\tau^2 \\ A_E &= -\frac{g}{6} D'_{E0}\,\Omega_A\tau^3 \end{aligned}\right\} . \tag{58}$$

The problem becomes the determination of A_1, A_2, A_3, and A_4 from a large set of redundant measurements $V_N(k)$ and $V_E(k)$. A_N and A_E are not needed because they differ from A_4 and A_2, respectively, only by a known constant multiplier.

2. The Usual Least Square Algorithm [6,7,8]

Measurement Equations (55) and (56) are in the form of three-term third-order polynomials in k. A conventional set of least square regression formulas are available in textbooks in statistical mathematics for determining the coefficients. Applying conventional formulas to our problem, the coefficients are determined from

$$\begin{bmatrix} A_1 \\ A_2 \\ A_N \end{bmatrix} = C^{-1} \begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \end{bmatrix} \tag{59}$$

and

$$\begin{bmatrix} A_3 \\ A_4 \\ A_E \end{bmatrix} = C^{-1} \begin{bmatrix} Z_1 \\ Z_2 \\ Z_3 \end{bmatrix} \tag{60}$$

where

$$Y_j = \sum_{k=1}^{N} k^j V_N(k) \qquad j = 1 \sim 3 \tag{61}$$

$$Z_j = \sum_{k=1}^{N} k^j V_E(k) \qquad j = 1 \sim 3 \tag{62}$$

$$C = \begin{bmatrix} C_2 & C_3 & C_4 \\ C_3 & C_4 & C_5 \\ C_4 & C_5 & C_6 \end{bmatrix} \tag{63}$$

and

$$C_i = \sum_{k=1}^{N} k^i \qquad i = 2 \sim 6 \quad . \tag{64}$$

These formulas have been used for platform self-alignment as well as their applications and are applicable to any three-term third-order polynomial. The algorithm can be written either for batch processing or for sequential processing.

3. A New Least Square Algorithm

Examining Equations (55) through (58), it is seen that they can be expressed in the following equivalent form:

$$V_N(k) = A_1 k + A_2 k^2 + uA_4 k^3 + n_N(k) \tag{65}$$

$$V_E(k) = A_3 k + A_4 k^2 - uA_2 k^3 + n_E(k) \tag{66}$$

where

$$u = -\frac{\Omega_A \tau}{3} \quad . \tag{67}$$

Notice that some of the coefficients are correlated deterministically. Can this property be employed to improve the estimation accuracy? The answer is in the affirmative.

Let us study a more general case

$$\left.\begin{aligned} V_K &= Ak + Bk^2 + uDk^3 + n_K \\ V'_K &= Ck + Dk^2 + vBk^3 + n'_K \end{aligned}\right\} \tag{68}$$

where V_K and V'_K are measurements made at sampling instants $k = 1 \sim N$, and A, B, C, and D are parameters to be estimated. A cost function is chosen as follows:

$$I = \sum_{k=1}^{N} \left\{ V_K - (Ak + Bk^2 + uDk^3) \right\}^2 + \sum_{k=1}^{N} \left\{ V'_K - (Ck + Dk^2 - uBk^3) \right\} \quad . \tag{69}$$

The cost function is to be minimized by an optimum selection of A, B, C, and D. Setting

$$\frac{\partial I}{\partial A} = \frac{\partial I}{\partial B} = \frac{\partial I}{\partial C} = \frac{\partial I}{\partial D} = 0 \quad . \tag{70}$$

a set of four algebraic equations are obtained which can be combined into a single matrix equation:

$$\begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} = G \begin{bmatrix} A \\ B \\ C \\ D \end{bmatrix} \tag{71}$$

where

$$G = \begin{bmatrix} C_2 & C_3 & 0 & uC_4 \\ C_3 & C_4 + VC_6 & VC_4 & (v + u)C_5 \\ 0 & VC_4 & C_2 & C_3 \\ uC_4 & (u + v)C_5 & C_3 & C_4 + VC_6 \end{bmatrix} \tag{72}$$

$$C_i = \sum_{k=1}^{N} k^i \qquad i = 2 \sim 6 \tag{73}$$

and

$$\left.\begin{aligned} W_1 &= \sum_{k=1}^{N} kV_K, & W_2 &= \sum_{k=1}^{N} k^2(V_K + vkV'_K) \\ W_3 &= \sum_{k=1}^{N} kV'_K, & W_4 &= \sum_{k=1}^{N} k^2(V'_K + ukV_K) \end{aligned}\right\} . \tag{74}$$

The estimate of A, B, C, and D are obtained by inverting the G matrix, giving

$$\begin{bmatrix} A \\ B \\ C \\ D \end{bmatrix} = G^{-1} \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} . \tag{75}$$

Applying this result to our platform model, given by Equations (65) through (67), results in

$$\begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} = G \begin{bmatrix} A_1 \\ A_2 \\ A_3 \\ A_4 \end{bmatrix} \tag{76}$$

where

$$G = \begin{bmatrix} C_2 & C_3 & 0 & uC_4 \\ C_3 & C_4 - uC_6 & -uC_4 & 0 \\ 0 & -uC_4 & C_2 & C_3 \\ uC_4 & 0 & C_3 & C_4 + uC_6 \end{bmatrix} \tag{77}$$

$$C_i = \sum_{k=1}^{N} k^i \qquad i = 2, 3, 4, \text{ and } 6 \tag{78}$$

$$\left.\begin{aligned} W_1 &= \sum_{k=1}^{N} kV_N(k), \qquad W_2 = \sum_{k=1}^{N} k^2[V_N(k) - ukV_E(k)] \\ W_3 &= \sum_{k=1}^{N} kV_E(k), \qquad W_4 = \sum_{k=1}^{N} k^2[V_E(k) + ukV_N(k)] \end{aligned}\right\}. \tag{79}$$

Values of A_i, $i = 1 \sim 4$, are given by

$$\begin{bmatrix} A_1 \\ A_2 \\ A_3 \\ A_4 \end{bmatrix} = G^{-1} \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} . \tag{80}$$

Notice that the measurements $V_N(k)$ and $V_E(k)$ contribute to the estimation via the computation of W_i, $i = 1 \sim 4$. Also, the property of the platform kinematics together with the choice of cost function result in the absence of C_5 in the computation.

The estimate for the initial state vector is obtained from A_i, $i = 1 \sim 4$, using Equations (57) and (58).

$$\begin{bmatrix} \theta_{NO} \\ \theta_{EO} \\ D'_{NO} \\ D'_{EO} \end{bmatrix} = \begin{bmatrix} \frac{A_3}{g\tau} \\ -\frac{A_1}{g\tau} \\ \frac{2A_4}{g\tau^2} \\ \frac{-2A_2}{g\tau^2} \end{bmatrix} . \tag{81}$$

Example 1

Consider a simple case having a single measurement equation

$$V_K = Ak + Bk^2 + Bk^3 + n_K . \tag{82}$$

Let A = 4, B = 2, and k = i ~ 10. The values of V_K are generated by adding noise n_K to the value of polynomial at each k. The values of n_K are taken from a table of normally distributed random numbers, having zero mean and a variance of one. These values are listed in Table 1.

TABLE 1. MEASUREMENT DATA FOR EXAMPLE 1

K	n_K	V_K
1	-1.276	6.724
2	-0.318	31.628
3	-1.377	82.623
4	2.334	178.334
5	-1.136	318.864
6	0.414	528.414
7	-0.494	811.506
8	1.048	1185.048
9	0.347	1656.347
10	0.637	2240.637

Both new and usual least square algorithms are used to estimate A and B from the data V_K, k = 1 ~ 10. The results are shown in Table 2, listing values of estimates and their percentage error as compared to true values. It is apparent that the new algorithm produces much more

TABLE 2. RESULTS FOR EXAMPLE 1

	A = 4	B = 2
Usual Algorithm N = 10	3.67969 8%	2.09863 5%
New Algorithm N = 10	3.95581 1%	2.00111 0.05%
New Algorithm N = 5, (1st 5)	3.87604 3.1%	2.00359 0.17%
New Algorithm N = 5, (2nd 5)	4.00342 0.086%	2.00059 0.025%

accurate estimates. The new algorithm gives better results even if fewer measurements are used. Appendix A contains the computer program for this example.

Example 2

Consider the following case of two measurement equations:

$$\left.\begin{aligned} V_K &= Ak + Bk^2 + Dk^3 + n_K \\ V'_K &= Ck + Dk^2 + Bk^3 + n'_K \end{aligned}\right\} . \qquad (83)$$

Let A = 4, B = 2, C = 3, D = 1, and k = 1 ~ 10. The values of V_K and V'_K are generated in a manner similar to that for Example 1. Table 3 lists the measurement data.

TABLE 3. MEASUREMENT DATA FOR EXAMPLE 2

K	n_K	n'_K	V_K	V'_K
1	-1.276	-1.218	5.724	4.782
2	-0.318	-0.799	23.682	25.201
3	-1.377	-1.257	55.623	70.743
4	2.334	-0.337	114.334	155.663
5	-1.136	0.642	193.864	290.642
6	0.414	-0.011	312.414	485.989
7	-0.494	0.364	468.506	756.364
8	1.048	0.037	673.048	1112.034
9	0.347	2.816	927.347	1568.816
10	0.637	0.563	1240.637	2130.563

Again, both new and usual least square algorithms are employed to estimate A, B, C, and D. The results are listed in Table 4, with the percentage error of each estimate estimated. Again, the new algorithm gives much better results. Appendix B presents the computer program for this example.

TABLE 4. RESULTS FOR EXAMPLE 2

	A = 4	B = 2	C = 3	D = 1
Usual Algorithm	3.61133 9.7%	2.08984 4.4%	2.07031 31.0%	1.2207 22.7%
New Algorithm	3.95093 1.2%	2.00272 0.14%	2.88883 3.7%	1.00099 0.1%

4. Sensitivity Consideration

It is expected that the new algorithm is less sensitive, as compared to the usual algorithm, with respect to erratic measurement data and to computation errors. The rationale is that the coupled parameters have a tendency to hold each other at their nominal values. Example 3 will show this effect.

Example 3

This example uses the same measurement model, same measurement data, and same computer programs as those used for Example 2. To observe the effect of erratic measurement data on estimates, the measurement V(10) is changed by 1% and a set of new estimate for A, B, C, and D is made using new and usual algorithms. To observe the effect of computation error on estimates, the value of V(10) is restored to its original value and the value of C_6 is changed by 0.01%. Another set of estimates are made using both algorithms. All estimates are listed in Table 5. Values of sensitivity in the table are calculated using the formula

$$S = \frac{\text{New estimate - Nominal estimate}}{\text{Nominal estimate}} . \qquad (84)$$

The result confirms the expectation that new algorithm has lower sensitivity.

Example 4

In this example a real world platform system is considered. The platform is of the class proposed for PERSHING II application. Twelve hundred and fifty pairs of measurement data were recorded at outputs of north and east accelerometers. The total sampling time is 240 seconds.

TABLE 5. SENSITIVITY COMPARISON FOR EXAMPLE 3

Condition		Nominal		1% Change in V(10)		0.01% Change in C_6	
Algorithm Used		Usual	New	Usual	New	Usual	New
A = 4	$\hat{A}$	3.61133	3.95093	5.41016	3.38013	2.64648	3.99084
	S	—	—	49.8%	-14.4%	-26.7%	1.0%
B = 2	$\hat{B}$	2.08984	2.00272	1.36035	2.00166	2.41113	2.00152
	S	—	—	34.9%	-0.05%	15.4%	-0.06%
C = 3	$\hat{C}$	2.07031	2.88883	2.07031	2.85061	0.13281	2.97167
	S	—	—	0.0%	-1.3%	-93.6%	2.87%
D = 1	$\hat{D}$	1.22070	1.00099	1.22070	1.01469	1.86183	1.00053
	S	—	—	0.0%	1.4%	52.5%	-0.05%

$\hat{A}$ - Estimate of A
S - Sensitivity

The quantities to be estimated are misalignments and platform drifts. New and usual least square algorithms are used for data reduction. The former consists of Equations (77) through (81) while the latter consists of Equations (59) through (64) and (87). The results of data reduction are shown in Table 6. To explore the sensitivity of both algorithms with respect to computation errors, a poor matrix inversion subroutine is used. When computations are done with double precision, both algorithms produce reasonable results. When computations are done with ordinary precision, the result from new algorithm is not reasonable, knowing the quality of the platform used. But the result from the usual algorithm is ridiculous regardless of the platform considered. The results show the superiority of the new algorithm.

TABLE 6. RESULT FOR EXAMPLE 4

	Ordinary Precision		Double Precision	
	New Algorithm	Usual Algorithm	New Algorithm	Usual Algorithm
θ_{NO}	1162.8 arcsec	4.736×10^{26} arcsec	-10.55 arcsec	-6.294 arcsec
θ_{EO}	695.6 arcsec	2.599×10^{26} arcsec	691.6 arcsec	693.0 arcsec
D'_{NO}	0.275 deg/hr	5.362×10^{36} deg/hr	13.306 deg/hr	13.19 deg/hr
D'_{EO}	-0.0017 deg/hr	2.824×10^{36} deg/hr	0.0775 deg/hr	0.038 deg/hr

The only disappointment in this example is that the exact values of misalignments and drifts were not available at the time of experiment, therefore a precise comparison of two results could not be made. To partially overcome this difficulty, theoretical error analyses are developed in Section VIII. These analyses will help to evaluate the quality of algorithms. Appendix C presents two computer programs used in this example.

Section VIII. THEORETICAL ERROR ANALYSES

The approach of theoretical error analyses used here is to develop analytical relationships relating the standard deviation of estimation error to the standard deviation of the noise. The analyses are done for new and usual least square algorithms.

1. Analysis for New Algorithm

Recall Equation (80) and define

$$H = G^{-1} \tag{85}$$

then

$$\begin{bmatrix} A_1 \\ A_2 \\ A_3 \\ A_4 \end{bmatrix} = H \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} . \tag{86}$$

H is a 4×4 matrix whose ij-element will be denoted by h_{ij}. Expanding the first row of Equation (86) and using the relationships in Equation (79), A_1 can be expressed as

$$\begin{aligned} A_1 &= \sum_{j=1}^{4} h_{ij} W_j \\ &= h_{11} \sum_k kV_N(k) + h_{12} \left\{ \sum_k k^2 V_N(k) - u \sum_k k^3 V_E(k) \right\} \\ &\quad + h_{13} \sum_k kV_E(k) + h_{14} \left\{ \sum_k k^2 V_E(k) + u \sum_k k^3 V_N(k) \right\} . \end{aligned} \tag{87}$$

The error in A_1 is caused by errors in $V_N(k)$ and $V_E(k)$ which are denoted by $e_N(k)$ and $e_E(k)$, respectively. Let e_1 represent the error of A_1, then from Equation (87) we can get

$$e_1 = h_{11} \sum_k k e_N(k) + h_{12}\left\{\sum_k k^2 e_N(k) - u \sum_k k^3 e_E(k)\right\}$$

$$+ h_{13} \sum_k k e_E(k) + h_{14}\left\{\sum_k k^2 e_E(k) + u \sum_k k^3 e_N(k)\right\}$$

$$= \sum_k \left\{\left(h_{11}k + h_{12}k^2 + h_{14}uk^3\right)e_N(k)\right.$$

$$\left. + \left(h_{13}k + h_{14}k^2 - h_{12}uk^3\right)e_E(k)\right\} \qquad (88)$$

which expresses the error in A_1 in terms of source errors.

Assume the following statistical properties for source errors:

a) Zero mean, i.e.,

$$< e_N(k) > = < e_E(k) > = 0 \qquad (89)$$

where "< >" denotes ensemble average

b) Uncorrelated between axes and from time to time, i.e.,

$$< e_N(i)e_N(j) > = < e_E(i)e_E(j) > = 0 \qquad i \neq j \qquad (90)$$

$$< e_N(i)e_E(j) > = 0 \qquad \text{all } i,j \qquad (91)$$

c) Equal and stationary variance, i.e.,

$$< e_N^2(k) > = < e_E^2(k) > = \sigma_e^2, \quad \text{a constant.} \qquad (92)$$

Taking the ensemble average over the square of Equation (88), applying previously mentioned error properties, and rearranging terms, we can obtain the error variance of A_1 as follows:

$$\sigma_{A_1}^2 = \sigma_e^2 \sum_k \left\{\left(h_{11}k + h_{12}k^2 + uh_{14}k^3\right)^2\right.$$

$$\left. + \left(h_{13}k + h_{14}k^2 - uh_{12}k^3\right)^2\right\} \quad . \qquad (93)$$

We shall digress for a moment to derive a number of relationships which will help to simplify the final expression. Substituting details

of $V_N(k)$ and $V_E(k)$ as given in Equations (65) and (66) into Equation (67) and regrouping terms,

$$A_1 = A_1 \sum_k k\left(h_{11}k + h_{12}k^2 + uh_{14}k^3\right)$$

$$+ A_2 \sum_k k^2\left(h_{11}k + h_{12}k^2 + uh_{14}k^3\right) - uk^3\left(h_{13}k + h_{14}k^2 - uh_{12}k^3\right)$$

$$+ A_3 \sum_k k\left(h_{13}k + h_{14}k^2 - uh_{12}k^3\right)$$

$$+ A_4 \sum_k k^2\left(h_{13}k + h_{14}k^2 - uh_{12}k^3\right) + uk^3\left(h_{11}k + h_{12}k^2 + uh_{14}k^3\right) \tag{94}$$

comparing both sides of Equation (94) shows that the coefficient of A_1 should be 1, and those of A_2, A_3, and A_4 should be zero. Therefore we get the relationships:

$$\left.\begin{aligned}
&\sum_k k\left(h_{11}k + h_{12}k^2 + uh_{14}k^3\right) = 1 \\
&\sum_k \left\{k^2\left(h_{11}k + h_{12}k^2 + uh_{14}k^3\right) - uk^3\left(h_{13}k + h_{14}k^2 - uh_{12}k^3\right)\right\} = 0 \\
&\sum_k k\left(h_{13}k + h_{14}k^2 - uh_{12}k^3\right) = 0 \\
&\sum_k \left\{k^2\left(h_{13}k + h_{14}k^2 - uh_{12}k^3\right) + uk^3\left(h_{11}k + h_{12}k^2 + uh_{14}k^3\right)\right\} = 0
\end{aligned}\right\} \tag{95}$$

Return to our analysis of error variance and apply the relationships of Equation (95) to Equation (93). The result is a very neat expression,

$$r_{11}^2 = \frac{\sigma_{A_1}^2}{\sigma^2} = h_{11} \quad , \tag{96}$$

where η_1^2 is the normalized error variance for A_1. The normalization is done with respect to the variance of source error.

In the similar manner, the normalized error variances for A_2, A_3, and A_4 are obtained as

$$\eta_2^2 = \frac{\sigma_{A_2}^2}{\sigma_e^2} = h_{22} \tag{97}$$

$$\eta_3^2 = \frac{\sigma_{A_3}^2}{\sigma_e^2} = h_{33} \tag{98}$$

$$\eta_4^2 = \frac{\sigma_{A_4}^2}{\sigma_e^2} = h_{44} \quad . \tag{99}$$

Notice that values of h_{ii}, $i = 1 \sim 4$, depend solely on N, the number of measurements, and u, the correlation parameter. Therefore, normalized error variances η_i^2, $N = 1 \sim 4$, are independent of measurement data, but depend on the kinematics of the platform misalignment.

By taking the square roots of Equations (96) through (99), equations for normalized standard deviation of estimates are obtained as

$$\eta_i = \sqrt{h_{ii}} \quad , \quad i = 1 \sim 4 \quad , \tag{100}$$

another very neat expression.

Equation (100) is very useful in several ways. For a given set of error standard deviations of the source and the number of measurements, it can be used to estimate the error standard deviations of estimates. However, for a given set of source error standard deviations and a set of prescribed standard deviations for estimates, it can be used to determine the minimum number of measurements needed. Finally, knowing the error standard deviation of the estimate and the number of measurements, the equation can be used to determine the standard deviation of source error, a tool for identification.

2. Analysis for Usual Algorithm

Recall Equations (59) and (60), and define

$$Q = C^{-1} \quad . \tag{101}$$

Q is a 4×4 matrix whose ij-element will be denoted by q_{ij}. Adopting an approach of derivation similar to that for the new algorithm, the expression of normalized error variances for estimates of A_i, $i = 1 \sim 4$, can be obtained as

$$\eta_i^2 = q_{ii} \quad , \quad i = 1 \sim 4 \quad . \tag{102}$$

Similarly, the normalized standard deviation expression is given by

$$\eta_i = \sqrt{q_{ii}} \quad , \quad i = 1 \sim 4 \quad . \tag{103}$$

In this case, η_i depends only on N, the number of measurements, but not on the correlation parameter.

3. Comparison

For a given measurement condition the accuracy of estimates produced by new and usual algorithms can be compared by comparing their standard deviations. Direct comparison of Equation (100) and Equation (103) in their literal forms is difficult. A numerical example will be used to demonstrate the superiority of new algorithm.

Example 5

Consider the same platform alignment problem of Example 4. Figures 8 and 9 show plots of normalized standard deviations as functions of N, the number of measurement. Axes of the plots are in log-scale.

For N = 1250, the new algorithm gives

$$\left.\begin{aligned} \eta_{\theta_{NO}} &= \eta_{\theta_{EO}} = 0.24 \times 10^{-7} \\ \eta_{D'_{NO}} &= \eta_{D'_{EO}} = 0.26 \times 10^{-13} \end{aligned}\right\} \quad , \tag{104}$$

while the usual algorithm gives

$$\left.\begin{aligned} \eta_{\theta_{NO}} &= \eta_{\theta_{EO}} = 0.15 \times 10^{-6} \\ \eta_{D'_{NO}} &= \eta_{D'_{EO}} = 0.94 \times 10^{-12} \end{aligned}\right\} \quad . \tag{105}$$

Comparing Equation (104) to Equation (105), the superiority of new algorithm is evident.

Appendix D contains the computer program used for providing plotting data for Figures 8 and 9.

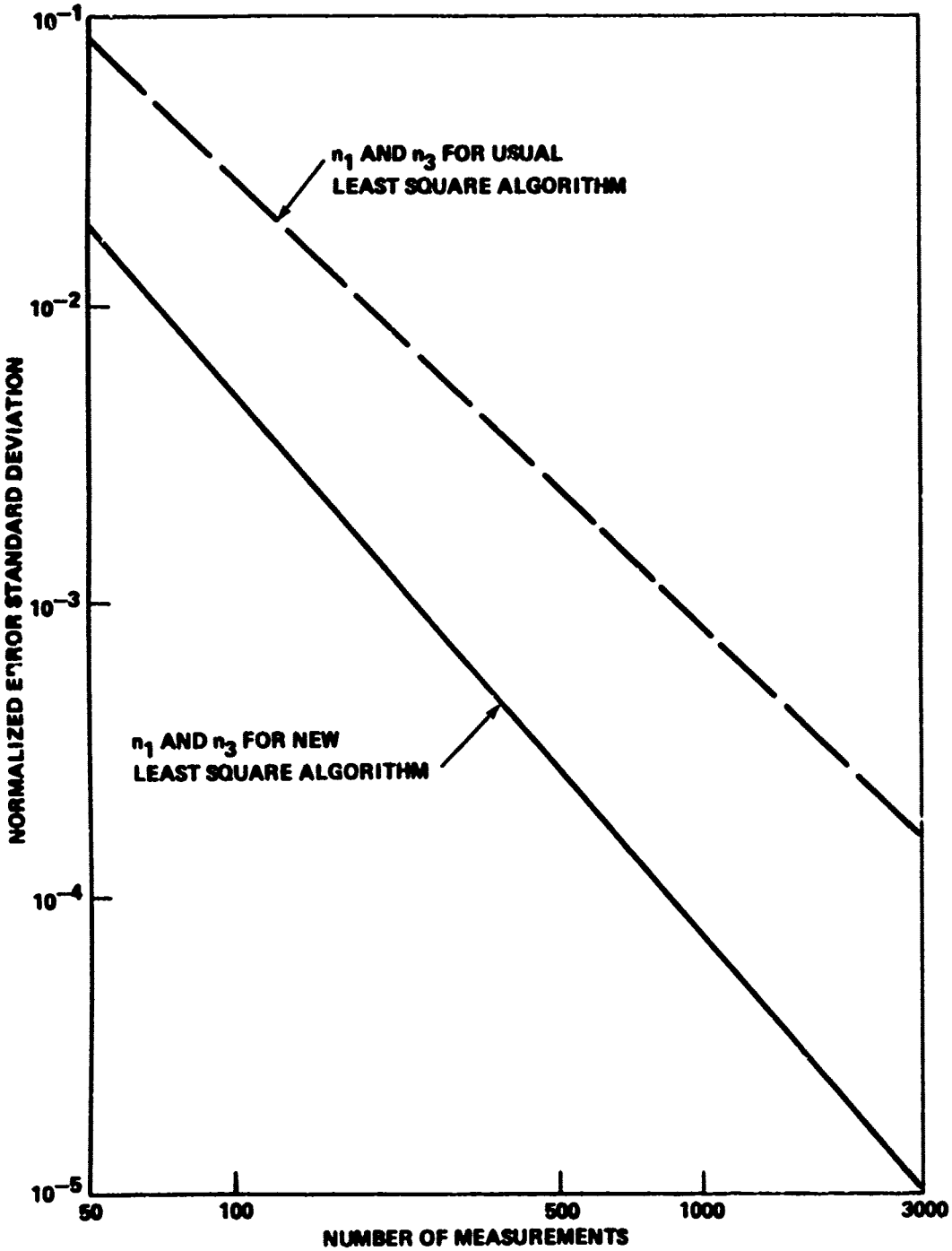


Figure 8. Normalized error standard deviation for Example 5.

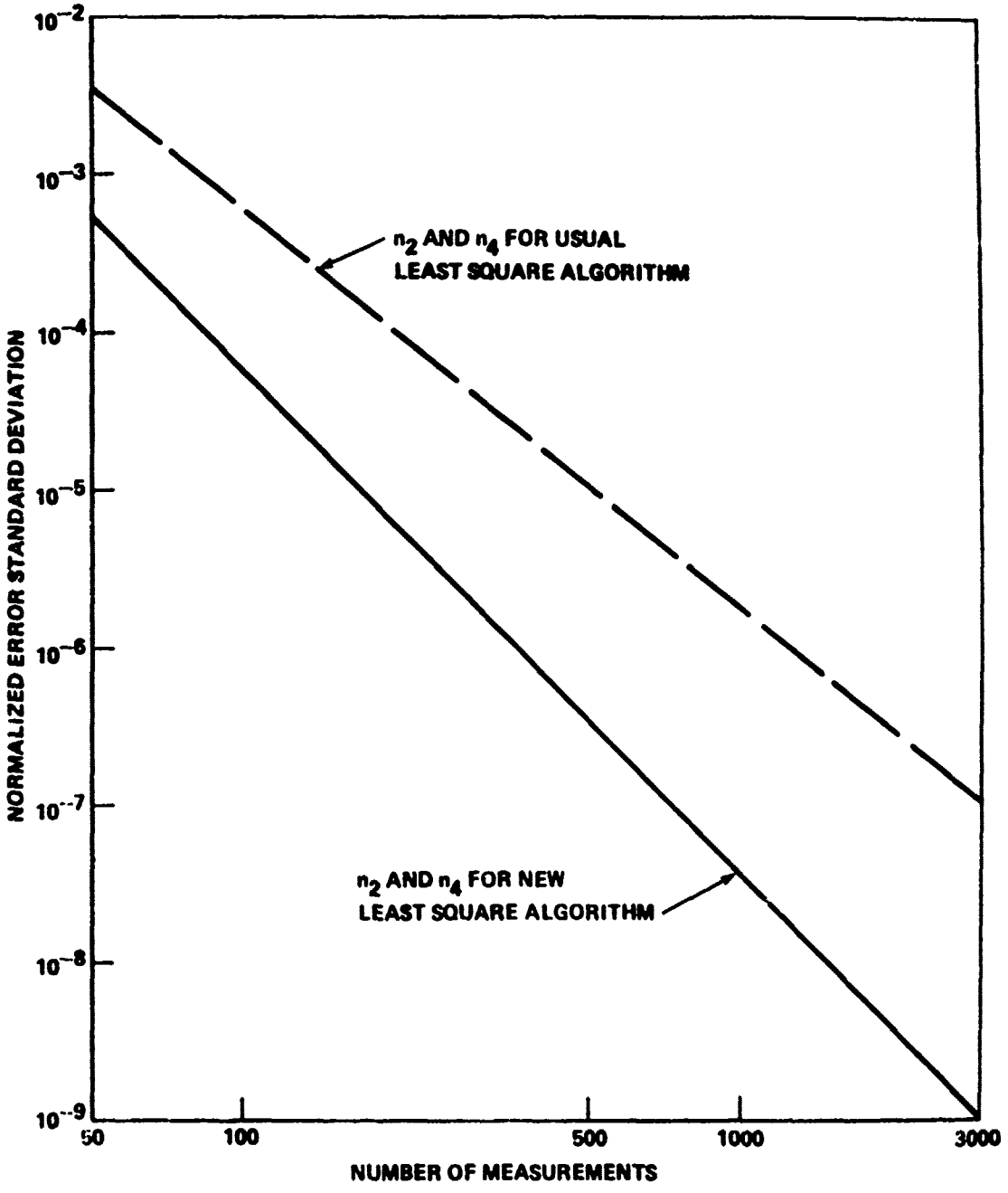


Figure 9. Normalized error standard deviation for Example 5.

Section IX. RECOMMENDED FURTHER STUDY

There are several problems which deserve further study. Solution to these problems will allow a truly optimum implementation of the IMU self-alignment systems.

Statistical theory shows that more accurate estimates are obtained with larger N, the number of measurements. However, computer error analysis shows that larger N results in more computer error because more computation is involved. It is desirable to choose an N such that the total error is at its minimum. A method for making such a choice is yet to be developed.

Even though the new least square algorithm is less sensitive to computation errors as compared to the usual algorithm, it is still desirable to keep computation errors as small as possible, expecially those occurring during matrix inversion. Notice that the G matrix of Equation (77) is symmetric and has four zero elements. This special form may allow the development of a matrix inversion subroutine which is more efficient in computation accuracy and computation time.

The new algorithm reported here is given in the form of batch process. This algorithm can be modified to become a sequential process or a hybrid process which is a semi-batch-semi-sequential process.

It is desirable to verify the analytically predicted superiority of the new least square algorithm for platform alignment by a precision hardware IMU which can be calibrated for experimental comparison.

The analytic results obtained from this study provide insights for coarse alignment which is required prior to the fine alignment. The coarse alignment can also be performed automatically and rapidly with the aid of the computer already available for fine alignment. It will be interesting to explore the possibility of a combined coarse and fine alignment using the same equipment and giving a best overall alignment result.

Section X. CONCLUSIONS

An original contribution of this study is the development and analysis of a new least square regression algorithm specially for the self-alignment of IMU systems. It was shown experimentally, as well as analytically, that this new algorithm is superior to the usual algorithm in accuracy and in sensitivity.

Although the new algorithm was originally intended for the self-alignment of a gimbaled platform, it can be used for the alignment of a strapdown platform as well, with some minor modifications. It can also be used for an IMU consisting of electro-optical sensors, because the underlying kinematic principle is similar.

Other results of this study include thorough derivation for drift equations, misalignment equations, and the gyrocompassing equation. Several self-alignment concepts were reviewed and discussed using the analytic foundation developed. Five examples were developed to help in confirming the theoretical prediction.

Several areas deserving further investigation were recommended. The solution to these areas will allow a truly optimum implementation of IMU self-alignment systems.

REFERENCES

1. Hung, J. C., State-of-the-Art Self-Alignment Techniques for Fixed Base Inertial Platforms, Report for Task Order 72-625, US Army Missile Command, Redstone Arsenal, Alabama, December 1973.

2. White, H. V., Servo Analysis of an Inertial Platform, Report No. RG-TR-63-3, US Army Missile Command, Redstone Arsenal, Alabama, July 1963.

3. Pitman, G., Editor, Inertial Guidance, John Wiley and Sons, New York, 1962.

4. Hung, J. C., A Study of Several Estimation Techniques for Gyrocompassing, A Report for Task Order 74-237, US Army Missile Command, Redstone Arsenal, Alabama, May 1974.

5. Kalman, R. E., "A New Approach to Linear Filtering and Prediction Problems," Trans. ASME, J. Basic Eng., Vol. 82D, 1960, pp. 34-45.

6. Hildebrand, F. B., Introduction to Numerical Analysis, McGraw-Hill, New York, 1956.

7. Ralston, A., A First Course in Numerical Analysis, McGraw-Hill, New York, 1965.

8. Burr, I. W., Applied Statistical Methods, Academic Press, New York, 1974.

9. Hamming, R. W., Numerical Methods for Scientists and Engineers, McGraw-Hill, New York, 1962.

10. Cannon, R. H., "Alignment of Inertial Guidance Systems by Gyrocompassing — Linear Theory," Journal of the Aerospace Science, Vol. 28, No. 11, 1961, pp. 885-895 and 912.

11. Kayton, M., and Fried, W. R., Avionic Navigation Systems, John Wiley and Sons, New York, 1969.

12. Danik, B., Gyrocompass Alignment of Inertial Platform, Document No. KD-73-9, The Singer Co., Kearfott Division, 6 February 1973.

Appendix A. COMPUTER PROGRAMS FOR EXAMPLE 1

(IN BASIC)

```
10  READ C2,C3,C4,C5,C6
20  DATA 385,3025,25333,220825.,1.97840E+06
40  LET Q1=C2
41  LET Q2=C3+C4
42  LET Q3=C4+2*C5+C6
50  LET D=Q1*Q3-Q2*Q2
60  LET M1=Q3/D
61  LET M2=-Q2/D
62  LET M3=M2
63  LET M4=Q1/D
100  DIM V[10]
110  FOR I=1 TO 10
120  READ V[I]
130  PRINT "V("I")="V[I]
140  NEXT I
150  DATA 6.724,31.6288,82.623,178.334,318.864
160  DATA 528.414,811.506,1185.05,1656.35,2240.64
180  LET X1=0
190  LET X2=0
200  FOR J=1 TO 10
210  LET X1=J*V[J]+X1
220  LET X2=J*J*(1+J)*V[J]+X2
230  NEXT J
260  LET A=M1*X1+M3*X2
270  LET B=M3*X1+M4*X2
280  PRINT "A="A,"B="B
290  PRINT "TRUE VALUES ARE:  A=4     B=2     C=B=2"
300  END
```

```
10  READ C2,C3,C4,C5,C6
20  DATA 385,3025,25333,220625,,1.97840E+06
105  REM  COMPUTATION OF K1 TO K9
110  LET D=C2*(C4*C6-C5*C5)-C3*(C3*C6-C4*C5)+C4*(C3*C5-C4*C4)
140  LET K1=(C4*C6-C5*C5)/D
150  LET K2=(C4*C5-C3*C6)/D
160  LET K3=(C3*C5-C4*C4)/D
170  LET K4=K2
180  LET K5=(C2*C6-C4*C4)/D
190  LET K6=(C3*C4-C2*C5)/D
200  LET K7=K3
210  LET K8=K6
220  LET K9=(C2*C4-C3*C3)/D
300  REM  READ IN OBSERVATIONS
305  DIM V[10]
310  FOR I=1 TO 10
320  READ V[I]
330  PRINT "V("I")="V[I]
340  NEXT I
350  DATA 6.724,31.682,82.623,178.334,318.864
360  DATA 528.414,811.506,1185.05,1656.35,2240.64
400  REM WITH PRECOMPUTED CONSTANTS, DATA PROCESSING BEGINS HERE.
405  LET X1=0
406  LET X2=0
407  LET X3=0
410  FOR J=1 TO 10
420  LET X1=J*V[J]+X1
430  LET X2=J*J*V[J]+X2
440  LET X3=J*J*J*V[J]+X3
450  NEXT J
500  LET A=K1*X1+K2*X2+K3*X3
510  LET B=K4*X1+K5*X2+K6*X3
520  LET C=K7*X1+K8*X2+K9*X3
530  PRINT "A="A,"B="B,"C="C
550  PRINT "TRUE VALUES ARE:  A=4     B=2     C=B=2"
600  END
```

Appendix B. COMPUTER PROGRAMS FOR EXAMPLES 2 AND 3

(IN BASIC)

```
10  PRINT "LEAST SQ. ALGO. USING PARAMETER CORRELATION"
12  LET C2=385
13  LET C3=3025
14  LET C4=25333
15  LET C5=220825.
16  LET C6=1.97840E+06
18  PRINT
20  DIM V[10]
30  FOR I=1 TO 10
40  READ V[I]
60  NEXT I
70  DATA 5.724,23.682,55.623,114.334,193.864
72  DATA 312.414,466.506,673.048,927.347,1240.64
80  DIM U[10]
90  FOR J=1 TO 10
100  READ U[J]
120  NEXT J
130  DATA 4.782,25.201,70.743,155.668,290.642
132  DATA 485.989,756.364,1112.04,1568.82,2130.56
200  DIM W[4,1]
210  LET W[1,1]=W[2,1]=W[3,1]=W[4,1]=0
240  FOR K=1 TO 10
250  LET W[1,1]=W[1,1]+K*V[K]
260  LET W[2,1]=W[2,1]+K*K*(V[K]+K*U[K])
270  LET W[3,1]=W[3,1]+K*U[K]
280  LET W[4,1]=W[4,1]+K*K*(U[K]+K*V[K])
290  NEXT K
400  DIM G[4,4]
401  LET G[1,1]=G[3,3]=C2
402  LET G[1,2]=G[2,1]=G[3,4]=G[4,3]=C3
403  LET G[1,3]=G[3,1]=0
404  LET G[1,4]=G[4,1]=G[2,3]=G[3,2]=C4
405  LET G[2,2]=G[4,4]=C4+C6
406  LET G[2,4]=G[4,2]=2*C5
460  DIM H[4,4]
470  MAT H=INV(G)
500  DIM X[4,1]
505  PRINT "X VECTOR"
506  PRINT
510  MAT X=H*W
520  MAT  PRINT X
540  PRINT "TRUE VALUES:  X1=4  X2=2  X3=3  X4=1"
600  END
```

```
10  PRINT "THE USUAL LEAST SQ. ALGO."
12  PRINT
20  DIM V[10]
30  FOR I=1 TO 10
40  READ V[I]
60  NEXT I
70  DATA 5.724,23.682,55.623,114.334,193.864
72  DATA 312.414,468.506,673.048,927.347,1240.64
80  DIM U[10]
90  FOR J=1 TO 10
100  READ U[J]
120  NEXT J
130  DATA 4.782,25.201,70.743,155.663,290.642
132  DATA 485.989,756.364,1112.04,1568.82,2130.56
200  READ C2,C3,C4,C5,C6
210  DATA 385,3025,25333,220825.,1.97840E+06
220  LET D=C2*(C4*C6-C5*C5)-C3*(C3*C6-C4*C5)+C4*(C3*C5-C4*C4)
230  LET K1=(C4*C6-C5*C5)/D
240  LET K2=K4=(C4*C5-C3*C6)/D
250  LET K3=K7=(C3*C5-C4*C4)/D
270  LET K5=(C2*C6-C4*C4)/D
280  LET K6=K8=(C3*C4-C2*C5)/D
310  LET K9=(C2*C4-C3*C3)/D
320  LET X1=X2=X3=0
350  FOR I=1 TO 10
360  LET X1=X1+I*V[I]
370  LET X2=X2+I*I*V[I]
380  LET X3=X3+I*I*I*V[I]
390  NEXT I
400  LET Y1=Y2=Y3=0
430  FOR J=1 TO 10
440  LET Y1=Y1+J*U[J]
450  LET Y2=Y2+J*J*U[J]
460  LET Y3=Y3+J*J*J*U[J]
470  NEXT J
480  LET A=K1*X1+K2*X2+K3*X3
490  LET B=K4*X1+K5*X2+K6*X3
500  LET C=K7*X1+K8*X2+K9*X3
510  LET E=K1*Y1+K2*Y2+K3*Y3
520  LET F=K4*Y1+K5*Y2+K6*Y3
530  LET G=K7*Y1+K8*Y2+K9*Y3
540  PRINT "A="A,"B="B,"C="C
550  PRINT "E="E,"F="F,"G="G
560  PRINT
570  PRINT "TRUE VALUES ARE:"
580  PRINT "A=1","B=2","C=1"
590  PRINT "E=3","F=1","G=2"
600  END
```

Appendix C. COMPUTER PROGRAMS FOR EXAMPLE 3

(IN FORTRAN)

```
      PROGRAM MAIN(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
C*****LEAST SQUARE ALGORITHM USING PARAMETER CORRELATION

      INTEGER DELPN, DELPE
      DIMENSION C(5), G(4,4), DELPN(1250), DELPE(1250), VN(1250),
     1          VE(1250), W(4), X(4), VNOFF(1250), VEOFF(1250), H(4,4)
      DOUBLE PRECISION G, H, W, X

C*****SETTING UP C(2) TO C(6)    ( C(5) IS NOT USED. )
      DO 2 I=2,6
      C(I) = 0.0
      DO 4 K=1,1250
      FK=K
    4 C(I)=C(I)+FK**I
    2 CONTINUE
      WRITE(6,6) (I, C(I), I=2,6)
    6 FORMAT (1H1/// 5(10X,*C*,I1,*=*E20.12/))

C*****ESTABLISHING G-MATRIX
C           EARTH RATE = 7.29211E-05 RADIAN/SECOND
C           LATITUDE = 34.6425 DEGREES
C           SAMPLING PERIOD, TAU = 0.192 SECOND
C           U = -(Y-COMPONENT OF EARTH RATE)*TAU/3
C             = -(-7.29211E-05 * SIN34.6425) * 0.192 / 3
C             = 0.265294738 2E-05
      U=0.265294738 2E-05
      G(1,1)=C(2)
      G(1,2)=C(3)
      G(1,3)=0.0
      G(1,4)=U*C(4)
      G(2,1)=G(1,2)
      G(2,2)=C(4)-U*C(6)
      G(2,3)=(-U)*C(4)
      G(2,4)=0.0
      G(3,1)=0.0
      G(3,2)=G(2,3)
      G(3,3)=C(2)
      G(3,4)=C(3)
      G(4,1)=G(1,4)
      G(4,2)=0.0
      G(4,3)=G(3,4)
      G(4,4)=C(4)+U*C(6)
      WRITE ( 8) ((G(I, J), J=1,4), I=1,4)
    8 FORMAT (1H0,7X,*G-MATRIX*/4X,4E24.14/4X,4E24.14/
     1          4X,4E24.14/4X,4E24.14)

C*****GETTING  G = G-INVERSE  (IN-PLACE STORAGE)
      CALL MTXINV(G,4)
      WRITE(6,9) ((G(I,J), J=1,4), I=1,4)
    9 FORMAT (1H0/7X,*G-INVERSE*/4X,4E24.14/4X,4E24.14/
     1          4X,4E24.14/4X,4E24.14)
```

```
C**********FOR DEBUGGING USE ONLY
C           TO CHECK THE INVERSE OF THE INVERSE OF G
            DO 80 I=1,4
            DO 81 J=1,4
   81       H(I,J)=G(I,J)
   80       CONTINUE
            CALL TXINV(H,4)
            WRITE(6,82) ((H(I,J), J=1,4), I=1,4)
   82       FORMAT(1H0/7X,*G-INVERSE"S INVERSE*/4(4X,4E24.14/))
C**********DEBUGGING INFORMATION ENDS

C*****READ IN MEASURED DATA
       READ(5,10) (DELPN(I), DELPE(I), I=1,1250)
   10  FORMAT(10(2I4))
       WRITE(6,12) (DELPN(I), I=1,1250)
   12  FORMAT(1H1,10X,*DELPN(I), I=1 TO 1250*,//63(8X,20I6/))
       WRITE(6,13) (DELPE(I), I=1,1250)
   13  FORMAT(1H1,10X,*DELPE(I), I=1 TO 1250*,//63(8X,20I6/))

C*****COMPUTING VNOFF(I) AND VEOFF(I)  (IN NUMBER OF PULSES)
       VNOFF(1)=DELPN(1)+55
       VEOFF(1)=DELPE(1)+40
       DO 40 I=2,1250
       AUXN = DELPN(I) + 55
       AUXE = DELPE(I) + 40
       VNOFF(I)=VNOFF(I-1)+AUXN
   40  VEOFF(I)=VEOFF(I-1)+AUXE
       WRITE(6,42)
   42  FORMAT (1H1,10X,*I*,8X,*VNOFF(I)*,12X,*VEOFF(I)*)
       WRITE(6,44) (25*I,VNOFF(25*I), VEOFF(25*I), I=1,50)
   44  FORMAT(4X,I8,2E24.14)

C*****OBTAINING VN(I) AND VE(I) BY COORDIANTE TRANSFORMATION
C           (IN NUMBER OF PULSES)
C      RATH = 150 DEGREES = 2.617993878 RADIANS
       RATH=2.617993878
       SR=SIN(RATH)
       CR=COS(RATH)
       DO 14 I=1,1250
       VN(I) = VNOFF(I)*CR - VEOFF(I)*SR
   14  VE(I) = VEOFF(I)*CR + VNOFF(I)*SR
       WRITE(6,30)
   30  FORMAT (1H1,10X,*I*,10X,*VN(I)*,15X,*VE(I)*)
       WRITE(6,15) (25*I, VN(25*I), VE(25*I), I=1,50)
   15  FORMAT (4X,I8,2E24.14)

C*****COMPUTING W-VECTOR
       W(1)=0.0
       W(2)=0.0
       W(3)=0.0
       W(4)=0.0
       DO 16 K=1,1250
```

```
      FK=K
      W(1) = W(1) + FK*VN(K)
      W(2) = W(2) + FK*FK*(VN(K)-U*FK*VE(K))
      W(3) = W(3) + FK*VE(K)
   16 W(4) = W(4) + FK*FK*(VE(K)+U*FK*VN(K))
      WRITE(6,17) (I, W(I), I=1,4)
   17 FORMAT (1H1/// 4(14X,*W*I1*=*E20.12/))

C*****COMPUTING X-VECTOR
      DO 18 I=1,4
      X(I)=0.0
      DO 20 J=1,4
   20 X(I) = X(I) + G(I,J)*W(J)
   18 CONTINUE
      WRITE(6,22) (I, X(I), I=1,4)
   22 FORMAT (1H0// 4(14X,*X*I1*=*E20.12/))

C*****COMPUTING PLATFORM PARAMETERS
C           N-AXIS SCALE FACTOR  SFN=100441.
C           E-AXIS SCALE FACTOR  SFE=101712.
      SFN=100441.
      SFE=101712.
      TAU=0.192
      A=1./TAU
      B=2./TAU**2
      ZETANO=A*X(3)/SFN
      ZETAEO=(-A)*X(1)/SFE
      ORFTNO=B*X(4)/SFN
      ORFTEO=(-B)*X(2)/SFE
      WRITE(6,24) ZETANO, ZETAEO, ORFTNO, ORFTEO
   24 FORMAT (1H0//14X,*ZETANO=*E20.12*  RADIAN*//
     1           14X,*ZETAEO=*E20.12*  RADIAN*//
     1           14X,*ORFTNO=*E20.12*  RADIAN/SEC*//
     1           14X,*ORFTEO=*E20.12*  RADIAN/SEC*)
C*****END OF THE ESTIMATION PROGRAM
      END
```

```
      SUBROUTINE MTXINV(G,M)
C
C*****ON INPUT G IS G, ON OUTPUT G IS THE INVERSE OF G
      DIMENSION G(4,4)
      DOUBLE PRECISION G
      DO 140 K=1,M
      IF (G(K,K)) 10, 160, 10
   10 GZZ=1.0/G(K,K)
      DO 90 I=1,M
      IF (I-K) 20, 90, 60
   20 CONST=G(I,K)*GZZ
      DO 50 J=I,M
      IF (J-K) 30, 50, 40
   30 G(I,J)=G(I,J)+CONST*G(J,K)
      GO TO 50
   40 G(I,J)=G(I,J)-CONST*G(K,J)
   50 CONTINUE
      GO TO 90
   60 CONST=G(K,I)*GZZ
      DO 80 J=I,M
      IF (J-K) 170, 80, 70
   70 G(I,J)=G(I,J)-CONST*G(K,J)
   80 CONTINUE
   90 CONTINUE
      DO 110 J=K,M
      IF(K-J) 100, 110, 180
  100 G(K,J)=G(K,J)*GZZ
  110 CONTINUE
      DO 130 I=1,K
      IF (K-I) 190, 130, 120
  120 G(I,K)=(-G(I,K))*GZZ
  130 CONTINUE
      G(K,K)=GZZ
  140 CONTINUE
      DO 150 I=2,M
      J1=I-1
      DO 150 J=1,J1
  150 G(I,J)=G(J,I)
      RETURN
  160 WRITE(6,210)
      GO TO 200
  170 WRITE(6,220)
      GO TO 200
  180 WRITE(6,230)
      GO TO 200
  190 WRITE(6,240)
  200 CONTINUE
  210 FORMAT(4X,*ZERO DIAGONAL ELEMENT, BAD DATA*)
  220 FORMAT(4X,*ERROR IN INDEXING IN DOING DO 80*)
  230 FORMAT(4X,*ERROR IN INDEXING IN DOING DO 110*)
  240 FORMAT(4X,*ERROR IN INDEXING IN DOING DO 130*)
      END
```

```
      PROGRAM MAIN(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
C     THE USUAL LEAST SQUARE ALGORITHM
C       3-TERM 3RD ORDER POLYNOMIAL

      INTEGER DELPN, DELPE
      DIMENSION DELPN(1250), DELPE(1250), VN(1250), VE(1250),
     1          VNOFF(1250), VEOFF(1250), Q(3,3)
      DOUBLE PRECISION Q

C*****SETTING UP C2 TO C6
      C2=0.0
      C3=0.0
      C4=0.0
      C5=0.0
      C6=0.0
      DO 2 K=1,1250
      FK=K
      C2=C2+FK**2
      C3=C3+FK**3
      C4=C4+FK**4
      C5=C5+FK**5
    2 C6=C6+FK**6
      WRITE(6,4) C2, C3, C4, C5, C6
    4 FORMAT (1H1///10X,*C2=*E20.12/10X,*C3=*E20.12/
     1          10X,*C4=*E20.12/10X,*C5=*E20.12/10X,*C6=*E20.12)

C*****COMPUTING D AND Q1 TO Q9
      Q(1,1)=C2
      Q(1,2)=C3
      Q(1,3)=C4
      Q(2,1)=C3
      Q(2,2)=C4
      Q(2,3)=C5
      Q(3,1)=C4
      Q(3,2)=C5
      Q(3,3)=C6
      CALL SYMINV(Q,3)
      WRITE(6,6) ((Q(I,J), J=1,3), I=1,3)
    6 FORMAT(1H0/7X,*Q-INVERSE*/3(4X,3E24.14/))

C*****READ IN MEASURED DATA
      READ(5,10) (DELPN(I), DELPE(I), I=1,1250)
   10 FORMAT(10(2I4))
      WRITE(6,12) (DELPN(I), I=1,1250)
   12 FORMAT(1H1,10X,*DELPN(I), I=1 TO 1250*,//63(8X,20I6/))
      WRITE(6,13) (DELPE(I), I=1,1250)
   13 FORMAT(1H1,10X,*DELPE(I), I=1 TO 1250*,//63(8X,20I6/))

C*****COMPUTING VNOFF(I) AND VEOFF(I)  (IN NUMBER OF PULSES)
      VNOFF(1)=DELPN(1)+55.0
      VEOFF(1)=DELPE(1)+40.0
      DO 40 I=2,1250
```

```
      VNOFF(I)=VNOFF(I-1)+DELPN(I)+55.0
   40 VEOFF(I)=VEOFF(I-1)+DELPF(I)+40.0
      WRITE(6,42)
   42 FORMAT (1H1,10X,*I*,8X,*VNOFF(I)*,12X,*VEOFF(I)*)
      WRITE(6,44) (25*I,VNOFF(25*I), VEOFF(25*I), I=1,50)
   44 FORMAT(4X,I8,2E24.14)

C*****OBTAINING VN(I) AND VE(I) BY COORDIANTE TRANSFORMATION
C           (IN NUMBER OF PULSES)
C      BATH = 150 DEGREES = 2.617993878 RADIANS
       BATH=2.617993878
       SB=SIN(BATH)
       CB=COS(BATH)
       DO 14 I=1,1250
       VN(I) = VNOFF(I)*CB - VEOFF(I)*SB
   14  VE(I) = VEOFF(I)*CB + VNOFF(I)*SB
       WRITE(6,30)
   30  FORMAT (1H1,10X,*I*,10X,*VN(I)*,15X,*VE(I)*)
       WRITE(6,15) (25*I, VN(25*I), VE(25*I), I=1,50)
   15  FORMAT (4X,I8,2E24.14)

C      COMPUTING Y1, Y2, Y3, Z1, Z2, Z3
       Y1=0.0
       Y2=0.0
       Y3=0.0
       DO 17 I=1,1250
       FI=I
       Y1=Y1+FI*VN(I)
       Y2=Y2+FI*FI*VN(I)
   17  Y3=Y3+FI*FI*FI*VN(I)
       Z1=0.0
       Z2=0.0
       Z3=0.0
       DO 18 J=1,1250
       FJ=J
       Z1=Z1+FJ*VE(J)
       Z2=Z2+FJ*FJ*VE(J)
   18  Z3=Z3+FJ*FJ*FJ*VE(J)

C*****COMPUTING X1 TO X4
       X1=Q(1,1)*Y1+Q(1,2)*Y2+Q(1,3)*Y3
       X2=Q(2,1)*Y1+Q(2,2)*Y2+Q(2,3)*Y3
       X3=Q(1,1)*Z1+Q(1,2)*Z2+Q(1,3)*Z3
       X4=Q(2,1)*Z1+Q(2,2)*Z2+Q(2,3)*Z3
       WRITE(6,20) X1,X2,X3,X4
   20  FORMAT (1H1///14X,*X1=*E20.12/14X,*X2=*E20.12/
      1                14X,*X3=*E20.12/14X,*X4=*E20.12)

C*****COMPUTING PLATFORM PARAMETERS
C           N-AXIS SCALE FACTOR  SFN=100441.
C           E-AXIS SCALE FACTOR  SFE=101712.
       SFN=100441.
       SFE=101712.
       TAU=0.192
```

```
      A=1./TAU
      B=2./TAU**2
      ZETANO=A*X3/SFN
      ZETAEO=(-A)*X1/SFE
      DRFTNO=B*X4/SFN
      DRFTEO=(-B)*X2/SFE
      WRITE(6,24) ZETANO, ZETAEO, DRFTNO, DRFTEO
   24 FORMAT (1H0//14X,*ZETANO=*E20.12*  RADIAN*//
     1           14X,*ZETAEO=*E20.12*  RADIAN*//
     1           14X,*DRFTNO=*E20.12*  RADIAN/SEC*//
     1           14X,*DRFTEO=*E20.12*  RADIAN/SEC*)
C*****END OF THE FORTRAN PROGRAM
      END
```

```
      SUBROUTINE SYMINV(G,M)
C*****ON INPUT G IS G, ON OUTPUT G IS THE INVERSE OF G
      DIMENSION G(3,3)
      DOUBLE PRECISION G
      DO 140 K=1,M
      IF (G(K,K)) 10, 160, 10
   10 GZZ=1.0/G(K,K)
      DO 90 I=1,M
      IF (I-K) 20, 90, 60
   20 CONST=G(I,K)*GZZ
      DO 50 J=I,M
      IF (J-K) 30, 50, 40
   30 G(I,J)=G(I,J)+CONST*G(J,K)
      GO TO 50
   40 G(I,J)=G(I,J)-CONST*G(K,J)
   50 CONTINUE
      GO TO 90
   60 CONST=G(K,I)*GZZ
      DO 80 J=I,M
      IF (J-K) 170, 80, 70
   70 G(I,J)=G(I,J)-CONST*G(K,J)
   80 CONTINUE
   90 CONTINUE
      DO 110 J=K,M
      IF(K-J) 100, 110, 180
  100 G(K,J)=G(K,J)*GZZ
  110 CONTINUE
      DO 130 I=1,K
      IF (K-I) 190, 130, 120
  120 G(I,K)=(-G(I,K))*GZZ
  130 CONTINUE
      G(K,K)=GZZ
  140 CONTINUE
      DO 150 I=2,M
      J1=I-1
      DO 150 J=1,J1
  150 G(I,J)=G(J,I)
      RETURN
  160 WRITE(6,210)
      GO TO 200
  170 WRITE(6,220)
      GO TO 200
  180 WRITE(6,230)
      GO TO 200
  190 WRITE(6,240)
  200 CONTINUE
  210 FORMAT(4X,*ZERO DIAGONAL ELEMENT, BAD DATA*)
  220 FORMAT(4X,*ERROR IN INDEXING IN DOING DO 80*)
  230 FORMAT(4X,*ERROR IN INDEXING IN DOING DO 110*)
  240 FORMAT(4X,*ERROR IN INDEXING IN DOING DO 130*)
      END
```

Appendix D. COMPUTER PROGRAMS FOR EXAMPLE 5

(IN FORTRAN)

```
      PROGRAM MAIN(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
C*****TABLE OF NORMALIZED STANDARD DEVIATIONS FOR THE USUAL
C         AND THE NEW LEAST SQUARE ESTIMATION ALGORITHMS

      DIMENSION A(4,4), Q11(30), Q22(30), Q33(30),
     1          G(4,4), H11(30), H22(30), H33(30), H44(30)

      U=0.26529473P2E-05
      C2=0.0
      C3=0.0
      C4=0.0
      C5=0.0
      C6=0.0
      DO 20 N=1,30
      M=(N-1)*100+1
      I=N*100
      DO 30 J=M,I
      C2=C2+J**2.0
      C3=C3+J**3.0
      C4=C4+J**4.0
      C5=C5+J**5.0
   30 C6=C6+J**6.0

      A(1,1)=C2
      A(1,2)=C3
      A(1,3)=C4
      A(2,1)=C3
      A(2,2)=C4
      A(2,3)=C5
      A(3,1)=C4
      A(3,2)=C5
      A(3,3)=C6
      CALL MTXINV(A,3)
      Q11(N)=A(1,1)
      Q22(N)=A(2,2)
      Q33(N)=A(3,3)

      G(1,1)=C2
      G(1,2)=C3
      G(1,3)=0.0
      G(1,4)=U*C4
      G(2,1)=G(1,2)
      G(2,2)=C4-U*C6
      G(2,3)=(-U)*C4
      G(2,4)=0.0
      G(3,1)=0.0
      G(3,2)=G(2,3)
      G(3,3)=C2
      G(3,4)=C3
      G(4,1)=G(1,4)
```

```
      G(4,2)=0.0
      G(4,3)=G(3,4)
      G(4,4)=C4+U*C6
      CALL MTXINV(G,4)
      H11(N)=G(1,1)
      H22(N)=G(2,2)
      H33(N)=G(3,3)
   20 H44(N)=G(4,4)

      WRITE(6,40)
   40 FORMAT(1H1/10X,*N/100*,19X,*G11(N)*,24X,*H11(N)*,25X,*H33(N)*)
      WRITE(6,50) (N, G11(N), H11(N), H33(N), N=1,30)
   50 FORMAT(10X,I5,10X,E20.12,10X,E20.12,10X,E20.12)

      WRITE(6,60)
   60 FORMAT(1H1/10X,*N/100*,19X,*G22(N)*,24X,*H22(N)*,25X,*H44(N)*)
      WRITE(6,70) (N, G22(N), H22(N), H44(N), N=1,30)
   70 FORMAT(10X,I5,10X,E20.12,10X,E20.12,10X,E20.12)

      WRITE(6,80)
   80 FORMAT(1H1/10X,*N/100*,19X,*G33(N)*)
      WRITE(6,90) (N, G33(N), N=1,30)
   90 FORMAT(10X,I5,10X,E20.12)

      END
```

```
      SUBROUTINE MTXINV(G,M)
C
C*****ON INPUT G IS G. ON OUTPUT G IS THE INVERSE OF G
      DIMENSION G(4,4)
      DO 140 K=1,M
      IF (G(K,K)) 10, 160, 10
   10 GZZ=1.0/G(K,K)
      DO 90 I=1,M
      IF (I-K) 20, 90, 60
   20 CONST=G(I,K)*GZZ
      DO 50 J=I,M
      IF (J-K) 30, 50, 40
   30 G(I,J)=G(I,J)*CONST*G(J,K)
      GO TO 50
   40 G(I,J)=G(I,J)-CONST*G(K,J)
   50 CONTINUE
      GO TO 90
   60 CONST=G(K,I)*GZZ
      DO 80 J=I,M
      IF (J-K) 170, 80, 70
   70 G(I,J)=G(I,J)-CONST*G(K,J)
   80 CONTINUE
   90 CONTINUE
      DO 110 J=K,M
      IF(K-J) 100, 110, 180
  100 G(K,J)=G(K,J)*GZZ
  110 CONTINUE
      DO 130 I=1,K
      IF (K-I) 190, 130, 120
  120 G(I,K)=(-G(I,K))*GZZ
  130 CONTINUE
      G(K,K)=GZZ
  140 CONTINUE
      DO 150 I=2,M
      J1=I-1
      DO 150 J=1,J1
  150 G(I,J)=G(J,I)
      RETURN
  160 WRITE(6,210)
      GO TO 200
  170 WRITE(6,220)
      GO TO 200
  180 WRITE(6,230)
      GO TO 200
  190 WRITE(6,240)
  200 CONTINUE
  210 FORMAT(4X,*ZERO DIAGONAL ELEMENT. BAD DATA*)
  220 FORMAT(4X,*ERROR IN INDEXING IN DOING DO 80*)
  230 FORMAT(4X,*ERROR IN INDEXING IN DOING DO 110*)
  240 FORMAT(4X,*ERROR IN INDEXING IN DOING DO 130*)
      END
```